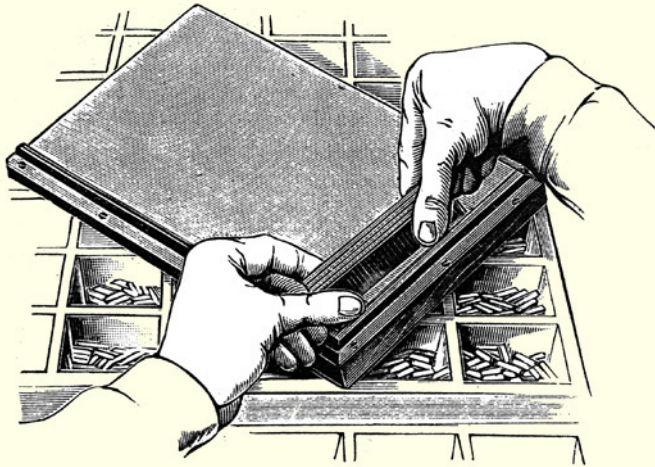


# THE TINY GUIDE

for

*PostScript Markup*



**DAVID  
BYRAM - WIGFIELD**



*Cappella Archive*

REVISED DECEMBER 2004

PDF Edition : Cappella Archive 2004

Copyright © 2003 David Byram-Wigfield

All rights reserved.

Acrobat and PostScript are trademarks of Adobe Systems Inc.,  
which may be registered in certain jurisdictions.

All other proprietary names are acknowledged as  
trademarks of their respective owners.

The Tiny Guide has been typeset by the methods described and the Portable Document Format Folios are made available for self-printing on condition that neither the originating portable format document nor the resulting hard-copy shall, by way of trade or otherwise, be lent, sold, hired-out, bound with other material, or circulated in any manner without the publisher's consent. The copyright notice must appear in all copies and in all materials generated by the contents.

Although placed in the public domain, the dictionary and the procedures described remain the intellectual property of the author. No representations are made about their suitability for any purpose, and no responsibility is assumed for any errors or inaccuracies therein nor any liability for incidental or consequential damages arising from their use.

British Library Cataloguing-in-Publication Data  
A catalogue record for the book is  
available from the British Library

ISBN 1-902918-13-4

*Cappella Archive*

Foley Terrace : Great Malvern : England

<http://www.cappella-archive.com>

### PostScript Markup Resources

The Tinydict and other resources may be found at:  
<http://www.cappella.demon.co.uk/tinyfiles/tinymenu.html>

### Selecting Examples

The PostScript procedures may be chosen by highlighting the relevant page and copying and pasting the selection into a text editor. Remove any footer text, page numbering, or navigation marks, and the comment lines beginning with a % must not linebreak.

*% iii %*

*% iv %*



## CONTENTS

Introduction	1
What is PostScript?	3
What is the Tinydict?	5
Typesetting Abilities	6
The File Structure	8
Paper Sizes, Page and Text Defaults	11
Text Styles and Orientation	13
Page Formats	15
Importing EPS files	21
Tinydict Procedures	23
Layout and Pagination	24
Columns	27
Running Text and Numbering Movement	29
Boxes and Circles	30
Linewrapping and Justification	32
Mark-up Codes	36
Default Typefaces	37
Resources	39
Printing Resources	41
Errors	43
PDF Resources	45
Re-encoding Typefaces	46
TinyTables	50
Usng Colour	55
Section Numbering	59
Setting Equations	60
Automatic Footnotes	61
Grep Markup	64
Automatic Hyphenation	65
Mark-up Syntax	66
PostScript Utilities	71
Printing a Book	84
Binding Instructions	86
Revisions	89
	90

## INTRODUCTION

The Tinydict had its origin in the series of articles I wrote in the 1990s for *Small Printer*, the journal of the British Printing Society. Its purpose is to mark-up ordinary text into the PostScript page description language for subsequent parsing into typeset book pages. This is done by downloading the coded file to a PostScript laser printer to produce two-sided hard copy ready for binding.

For archiving purposes, the files remain editable at all times; they are independent of computer systems and the complexities and up-grading vagaries of commercial typesetting software are avoided. By using the Adobe Acrobat, or a PostScript emulator such as Ghostscript, the marked-up files may also be converted into the Portable Document Format for on-screen proofing or Internet transmission for universal printing or viewing.

The mark-up process is quite simple. The text is placed between pairs of parentheses, otherwise called brackets (like these), or in computing jargon described as delimiters. They are inserted before and after the first and last word of each paragraph or an individual line. Each closing parenthesis is followed by a code letter which marks up the intervening text for typesetting. Most text editors can prefix and suffix the delimiters and chosen mark-up code very quickly, using the search and replace facility.

Any available PostScript font and style may be used at any size, and the Tinydict codes are meant to be self-explanatory. There are codes for setting text flush left or right, forcing justification, placing running headers and footers, coloured text, boxes and circles, and inserting up to five columns on a page. There are add-on resources for tables, equations, footnotes, pdf links, and even one for music notation.

I am especially grateful to Olavi Sakavi for devising the PostScript automatic hyphenation procedures used by the Tinydict and thereby solving one of the most complex problems in mark-up typesetting.

*David Byram-Wigfield*  
*Great Malvern, England*



## EQUIPMENT

The requirements for using the Tinydict Mark-up are:

A computer of almost any system and a text editor

AND

A PostScript printer and downloader.

OR

A software PostScript emulator  
(e.g. StyleScript or Ghostscript)

OR

The free Adobe Reader and a third party  
Portable Document Format distiller

OR

The full Adobe Acrobat for PDF conversion  
(use v3.0.2 on older PCs and classic Macs)

A word processor may be used if a text editor is not available, but files must be saved in TXT format before PDF conversion or downloading to a PostScript printer. A line unwrapping text editor is preferable because Unix systems return each on-screen line by means of a line feed; and PCs use a return followed by a linefeed . Each of these inserts extra spaces into PostScript marked-up copy.

The spaces may be removed by either unwrapping the text \ so that it runs off the right hand side of the monitor \ window or by inserting the PostScript ‘soft return’ of \ space – backslash – return at the end of each on-screen \ line as shown in this example. The returns have no relation \ to the appearance of the printed text on the page.

A range of PostScript editors and utilities is listed on page 79 and the latest revisions on page 84. Level 1 users should also uncheck ‘findfont’ in section 36 or an ‘undefined’ error may result.

Download the Tinydict and associated resources from:

<http://www.cappella.demon.co.uk/tinyfiles/tinymenu.html>

## WHAT IS POSTSCRIPT?

PostScript was developed in 1982–5 by John Warnock and Chuck Geschke of Adobe Systems Inc. as a written description of the text and images to be printed on a page. The script provides instructions similar to those for finding buried treasure; a line is drawn from a starting point so many paces north, so many east, etc. The paces, measured at seventy-two printer’s points to the inch, are interpreted by a special computing chip placed in a laser printer.

PostScript rapidly became the universal page description language that it is today when desktop printing software like PageMaker and Quark XPress was devised to convert the images on the screen into scripted recipes. Despite their popularity for art-work and graphic design, using them to typeset and edit a lengthy book over hundreds of on-screen pages still remains a daunting task.

There are other difficulties in using commercial typesetting programs. The generated PostScript scripts are impossible to edit without a return to the original software version, which makes archiving unreliable. Even worse, adjoining words and characters in the text are constantly moved closer or further apart in an attempt to ‘improve’ the appearance of the text on the page.

This typographical restlessness involves a software-generated combination of ‘tracking’ and ‘kerning’ with the movement coordinates often expressed to six decimal places (i.e. one millionth of an inch). Here are samples from some PostScript files:

- a. 104.849655 149.952377 moveto 0 0 32 0 0 (The orientation ) ts
- b. 8155(checking,)s 0 6261(or)m 259(e)s 6(v)k 3(en)k
- c. [1 0 0 1 52 433]e (does)t T [1 0 0 1 80.0421 433]e (so)t
- d. 300.046 50.6245 m (l)M 304.609 h -.525 0 3 26.062 (ndex)d

The complex digital barrier between the computer screen and the printer, makes most users unaware of the elegance, accuracy and efficiency of PostScript as a scripted printing language; requiring as it does only the simplest of text editors to send typesetting instructions directly to the printer interpreter.



## WHAT IS A MARK-UP?

In the 1980s, letter codes were used to mark-up the copy for computerized newspaper printing. The marks were of two kinds; a generalized command for a pre-determined editorial format (such as sub-headings), or a succession of specific codes required by the house style of body text, typeface, linespacing, and column width. These were often grouped into a single 'macro' for swifter keying.

The generalized mark-up is still used for typesetting scientific papers in a program such as T<sub>E</sub>X. An author types '\chapter' or '\footnote' at the relevant point in the text for it to be correctly set on the page.

Once desktop printing software enabled pages to be proofed on the computer screen, the mark-up method fell out of general favour until revived by the arrival of the HyperText Mark-up Language. This was devised for the creation of Internet web pages and uses fontsize formatting codes such as <H1>. . .</H1>; though the actual typeface read on screen may also be determined by the recipient.

The PostScript procedures described in these pages daisy-chain various instructions together to form a mark-up method for any operating system. The advantages are that the codes are simple; the typeset files are always editable, and may be distilled into the Portable Document Format for proofing or internet transmission.

## WHAT IS THE PORTABLE DOCUMENT FORMAT?

As already mentioned, commercial typesetting software produces its own idiosyncratic PostScript script to place text and images on the printed page. The resulting incompatibility between different operating systems and printing programs led Adobe Systems Inc. to develop the Portable Document Format, so that a PostScript file from one source could be merged with another. The process has the advantages of on-screen display; interpretation by most kinds of printer, laser or inkjet; and security against unauthorized access.

Marked-up files may be converted to the portable document format by using the Adobe Distiller, or by using the share or freeware listed at the end of this Guide.

## WHAT IS THE TINYDICT?

The Tinydict is a set of mark-up procedures for typesetting any text file into the PostScript page description language. The script will automatically paginate the copy into book-sized facing pages in folio, wire, or punched binding order, for printing on a PostScript laser printer or conversion into Portable Document Format.

The following formats and resources are available and pages type-set in one are easily converted to another by changing the formatting keyword. The 'A' references are for comparison only; alternative paper sizes such as north American letter, ledger, and tabloid, may be specified in the page formatting header.

1. 1upA5 sets single pages for proofs, ebooks, or re-imposition.
2. 1upA4 sets single pages for letters, posters, or re-imposition.
3. 1upA3 sets single pages for posters or artwork.
4. 2upA4 sets facing pages in 4-1: 2-3 sequence for folio binding.
5. 2upPP enables printer's pairs for sewn or stapled sections.
6. 2upLR sets left to right reading for on-screen pages.
7. 4upA4 imposes 4up for miniature proofs or pocketbooks.
8. 4upA3 imposes 4up for full-size work and turn folios.
9. 8booklet imposes an 8 page folding order on one sheet.
10. 8wide imposes a landscaped 8 page folding order on one sheet.
11. 8legal imposes 16 pages 4up on two sheets for pocketbooks.
12. 16nested imposes 16 pages 4up on two sheets for sewn sections.
13. Tinytiles imposes 16 consecutive miniature proofs on one A4 sheet.
14. Up to 5 columns may be set for brochures or news-sheets.

The Tinydict is pasted at the head of a text file and the script is marked-up by simple codes. Alternatively, the dictionary may be previously downloaded by unchecking 'startjob' in section 1. You may mark-up files of almost any length; the simplest being:

**%!PS**

**% paste the Tinydict here**

**1upA4**

**(This is some text, set left as either a single word, a line, or a paragraph. ) P**

**close**

Remember the space before the final ) Try it!



## TYPESETTING ABILITIES

Formats pages as 1, 2, 4, 8, or 16up in folios or sections.  
Sets text justified left, right, centered, forced, or full.  
Automatic hyphenation from a choice of dividing dictionaries.  
Sets text in up to five columns on a page.  
Sets coloured text in single words, lines, or paragraphs.  
Advances or reverses any whole or fractional distances.  
Auto-flows text from page to page, numbering as it goes  
Numbers pages with running headers, footers or folios.  
Enables folio or roman numbers for preliminary pages.  
Uses any available PostScript Type 1 or 3 typeface.  
Changes font styles, text sizes, and line-spacing at any time.  
Places EPS files and resizes them as required using Distiller.  
Creates in-line boxes and circles, vertical and horizontal rules.  
Creates coloured outlines and backgrounds.  
Chapter headings may be forced onto a fresh page.  
Has a simple but effective tabbing system.  
For wide range of additional resources, see section 37

## WHAT CAN'T IT DO?

The Cappella Archive Dividing Dictionary is a resource still under development. Users preferring the TeX hyphenation algorithms will find them on the Cappella Archive Web pages.

Text may be placed to the right and left of a graphic by using the 1st and 2nd column facility, but there is no irregular runaround available. Such lines of text may be adjusted individually. See section 20.

In-line changes of typeface or font size are easily made by using the 'S' or 'j' codes, although, when using full justification, the line may sometimes need some fine-tuning after proofing.

## A File Header

```
%!PS                                % use %!PS-Adobe-2.0 or 3.0 for DSC compatibility
%%Title: filename.ps
%%BoundingBox: 0 0 842 597          % A4 landscaped: use 792 612 US letter
%%Creator: byram@cappella-archiv.com
%%Date: 3 March 2003
%%EndComments                       % no colon needed

%%BeginProlog
                                % insert any instructions from the printing resource
%%IncludeResource: tinydict.ps      % paste here or uncheck 'startjob'
%%EndProlog

%%BeginSetup
/2upA4 { _Z                        % format name and definition brace: place an isolator
  A4                                % paper size from section 3: use 'a4' or 'letter' for level 1
  300 RM 72 FM 42 IM                % adjust defaults for alternative paper sizes
  headright tinydict begin         % landscape facing pages and open the dictionary
  0 0 translate                    % an optional x y register adjustment
  /margins { 470 TM 300 RM 0 BM 0 LM 0 IN } def % margin resetter
  /bodytext { 10 rom 12 LG MT } def % 10 pt roman on 12 pt linespacing
  /textbox { margins bodytext } def % resets defaults for each new page
  /numbering { footers } def       % or use 'headers': 'number folio' or leave empty.
  /chapter (Cappella Archive) def  % verso (left hand) running text
  /title (PostScript Markup) def   % recto (right hand) running text
  4pages p1                        % create folios: no space in '4pages': open first page
} def                               % all format definitions must be closed with a curly brace
%%EndSetup

%%BeginScript
front                              % prefix all the others with % to print the first side
%back                              % for folios and nested sections: remove % to print the reverse
%inner                             % for nested sections only: remove % to print the second first pass
%outer                             % for nested sections only: remove % to print the second reverse
15 PG                              % page number only needed if first page is more than 1
2upA4                              % open the page format
                                % The marked-up copy is inserted here %
%%EndScript

%%Trailer                          % DSC comment
close                              % print the page/s: close the file
%%EOF                              % end of file marker
```





## THE FILE STRUCTURE

### The Header

A PostScript file has four parts; a header of %% comments called document structuring conventions, or DSC for short; a Prolog with the necessary resources; a Set-up section organizing the required data; and a Script of instructions for placing the text and pictures on the page.

The Adobe numbers refer to the version of DSC being used, not to the level of PostScript. Some emulating software may reject a file if the Adobe-2.0, EndComments, and Page and Trailer markers are missing. BoundingBox values are also needed for some viewers and the last two paper size values should be swapped as shown for landscape orientation.

Elaborate Adobe DSC comments are really only necessary for press bureaux file exchange, imagesetter manipulation, and for colour separation. Apart from the minimalist header shown above, and the useful Include Resource instruction, they may be generally avoided by knife and fork typesetters.

The Tinydict marked-up files are *not* page independent, as there may be several book pages to each PostScript ‘page’ (i.e. sheet of paper). If you need page independence for re-imposition into traditional multi-page sections use the 1upA5 format and, after distilling, export the file through Adobe Acrobat as DSC page-compliant.

### The Prolog

The Prolog holds resources for the interpretation of the file and any printing instructions. The Tinydict should be pasted here unless ‘startjob’ in the dictionary header is unchecked and it is downloaded previously. The Tinydict will then remain in the printer, Distiller, or emulator memory until the device or software is quitted.

Instructions for manual feed, duplexing, or the number of copies, may be inserted in the Prolog for local printing, but must not be included if the file is to be distilled into a PDF document or sent for commercial printing. See the printing resources on page 42.

The ‘setpagedevice’ instruction is not recognized by level 1 printers and ‘a5’ or ‘letter’ should be substituted when necessary.

## The Set-up

The Tinydict includes 1, 2, 4, 8, and 16 page formats and the format name typed at the head of the script will automatically open the required pages. Nevertheless, a marked-up file should record all the details shown, if only to avoid conflict with Tinydict values altered for other work. The format Set-up allows global changes of pagination, such as 1up to 4up, and font, size, or linespacing to be made at any time. Formats are defined in section 9 for copying and pasting into the typesetting file and font selection is in section 36.

The ‘4pages’ example quoted provides a sequence of four library size book pages in folio folding order 4-1: 2-3. The facing page text box areas are positioned by the depth of the footer margin, the width of the inner margin, and the width of the text to the right margin RM. Notice that the RM value must be inserted twice; once to place the pages accurately and then as a textbox margin default.

A numbering style may be chosen with an incrementing figure at the page foot (confusingly also called a folio) or placed at the top or bottom of the page with some associated text in running headers or footers. Leave the braces empty if none of these is required.

The optional horizontal (x) or vertical (y) translation shift may more accurately register the front and back printed pages. Test by checking whether the left and right hand outer margins are the same width. Notice that ‘translate’ is placed after any headright landscaping so that a positive x moves margins right and y moves them upwards. To make the same movement using headleft they will need negative values.

### The Script

The four page folio opening commands are ‘front 2upA4’ for the first pass and an unchecked ‘back’ for the second on the reverse. Don’t try at first to print a lengthy book from a single file; divide the script into multiples of four pages, such as 16 or 32, and alter the PG page number to suit. The first page of a facing pair is always right hand (recto) and has an odd number. An exception is the 2screen pagination which reads from left to right. If laser printing, use the manual feed to keep an eye open for double paper feeds.



## **% 1. PROLOG %**

```
%%BeginProlog
% true 0 startjob          % level 2: uncheck the % to download independently
userdict begin           % sections 1 to 15 are placed in the interpreter user dictionary
/PW { /pw exch def } def 597 PW      % A4 defaults: for letter use 612 PW 792 PH
/PH { /ph exch def } def 842 PH
/languagelevel where { pop languagelevel 1 gt      % enable level 1 interpreters
{/setpaper {<< /PageSize [ pw ph ] >> setpagedevice } def } ifelse pop
{ a4 } ifelse } { a4 } ifelse } def                % substitute 'letter' if preferred
/pdfmark where { pop } { userdict /pdfmark /cleartomark load put } ifelse
%%EndProlog
```

Unchecking 'startjob' will download the dictionary into the PostScript printer, Distiller or emulator memory. Do not uncheck if the Tinydict is at the head of a file. The 'userdict begin' must have the matching 'end' in section 15.

Old level 1 devices cannot use 'startjob' or 'setpagedevice' and north American users should substitute 'letter' for the default paper size. See the printing resources for tray selection, multiple copies, and duplexing instructions.

The pdfmark enquiry is necessary for pdf documents or e-books which include bookmarks for on-screen display, yet may be also parsed by a PostScript printer or emulator.

## **% 2. ISOLATORS %**

```
/_Z { /defaults save def } def          % preserve existing condition
/ZZ { defaults restore } def           % restore previous state
/cleanup { clear cleardictstack } def
/close { showpage grestore cleanup ZZ } def % 'grestore' closes last 'gsave'
```

The abbreviated save-restore twins are placed at the beginning and end of a marked-up file to restore the normal state. They are also used either side of imported PS or EPS files to isolate any changes made in orientation, colour, scaling, translation, or definition, and restore the memory consumed by doing so.

Files conforming to the Document Structuring Conventions also use a save-restore pair to isolate each page, so that they may be selected at random and printed or imposed in any order. The final page is printed by 'close' and the printer defaults are reset by ZZ.

## **% 3. PAPER SIZES %**

```
/A5 { 420 PW 595 PH setpaper } def      % level 1 use 'a5' instead of 'setpaper'
/A4 { 597 PW 842 PH setpaper } def      % all sizes in points at 72 per inch
/A3 { 842 PW 1190 PH setpaper } def
/B4 { 729 PW 1032 PH setpaper } def
/LETTER { 612 PW 792 PH setpaper } def  % level 1 use 'letter' not 'setpaper'
/FOLIO { 612 PW 936 PH setpaper } def
/LEGAL { 612 PW 1008 PH setpaper } def  % level 1 use 'legal' not 'setpaper'
/TABLOID { 792 PW 1224 PH setpaper } def
```

When laser printing, do not vary the standard sizes by more than one percent or an error may result. Distiller, however, will allow custom sizes for e-books. You can substitute the format paper size with, say, '300 PW 400 PH setpaper' and adjust RM and TM to fit. Upper case characters define 'setpaper'; only use lower case for level 1 devices.

## **% 4. PAGE DEFAULTS %**

```
/FM { /fm exch def } def 72 FM          % footer margin = 1"
/OM { /om exch def } def 30 OM          % outer margins for brochures
/IM { /im exch def } def 42 IM          % 2up inner margins = 0.625" each
/NC { /nc exch def } def 2 NC           % default number of columns
/PG { /pg exch def } def 1 PG           % default first page number
/HY { /threshold exch def } def 5 HY    % default hyphenation strength
```

The footer margin FM positions the bottom of the textbox on the page and should remain a constant size, or book pages printed at a later date may not be level with previous ones. The inner margin IM determines how far the facing textboxes are apart. Increase to 50 IM for a wider spine margin for wired or punched binding and reduce to 36 for letter paper, to avoid gripper smear. The outer margin OM is used with FM for positioning a textbox on brochures.

The default first page number PG is set here and not in the textbox (or every page would have the same number). Similarly, NC determines the default number of columns on a page and any alternative will remain in force until another is selected.

The HY hyphenation threshold determines the automatic hyphenation intensity on a odd number scale of 1, 3, 5, 7, with 7 being the weakest. Use 1 HY for narrow columns. Don't invoke HY if the hyphenation resource is not in section 37.





## % 5. TEXTBOX DEFAULTS %

**/TM { /tm exch def } def 470 TM** % top margin = 6.5" (165mm) = text height  
**/BM { /bm exch def } def 0 BM** % bottom margin: default zero  
**/LM { /lm exch def } def 0 LM** % left margin: default zero  
**/RM { /rm exch def } def 300 RM** % right margin = 4.125" (100mm) = textwidth  
**/IN { /in exch def } def 0 IN** % text or box inset value  
**/LG { /lg exch def } def 12 LG** % default line spacing  
**/margins { 470 TM 300 RM 0 BM 0 LM 0 IN 2 NC } def** % for facing book pages

The four textbox margins dimension the text area on the page. The top and right margins, TM and RM, control the text height and width. Note that TM measures from the bottom textbox margin upwards, not downwards from the top. The right margin RM specifies the width of the textbox from left to right, (traditionally, on a book page it was 288 points wide and called the ‘measure’). It also positions the textbox on the verso (left-hand) of facing pages.

*If you change RM in the page format set-up you must change it after the margin defaults in the textbox as well.*

The bottom and left margins, BM and LM, are set at zero so that the text formatting remains constant if the paper sizes are altered or the textbox is moved. Their values are re-adjusted automatically for footnotes or indents. Text and boxes use values given to IN to determine the amount of inset or outset.

The margin values are grouped to reset the defaults for each new page. You may of course give FM, IM, TM, and RM any value you wish. Any on-page margin changes should be followed by a movement and the tenth of a point advance ‘MT’ is sufficient for the values to take immediate effect. A move to a fixed position on the page may be specified using TM, such as ‘70 TM MT’.

The margin values are always quoted in the page formats and the top and right margins are then amended to suit the chosen paper size. A default 12 point linespacing is defined here but this may be reset to any value in the bodytext.

## % 6. TEXT STYLES %

**/bodytext { 10 rom 12 LG MT } def** % default body text size, style, and linespacing  
**/textbox { margins bodytext } def** % resets defaults for a newpage  
**/quotetext { lg IN INSET 9 rom 11 LG MT } def** % reduce sizes and inset text  
**/QT { quotetext /textbox { margins quotetext } def } def** % quote style  
**/QQ { OUTSET 0 IN bodytext /textbox { margins bodytext } def } def**  
**/coltext { 10 LG 8 ss } def** % default column text style, size, and spacing  
**/numbering { footers } def 12 LG** % or ‘headers’, ‘number folio’, or leave empty.  
**/chapter (Cappella Archive) def** % running text recto  
**/title (The Tiny Markup ) def** % running text verso

The bodytext is the library style of 10 point roman on 12 point linespacing and the default typefaces must be left ungrouped in section 36. Alternative fonts and styles or sizes may be set in groups, and their collective title inserted in the bodytext definition.

The tenth of a point MT is needed to move text to the correct starting point. A minus T value will outdent single lines for margin notes or hanging capitals.

- QT** Other text styles may be compiled, like the quotetext shown here, which provides 9 point roman on 11 point linespacing, indented to the left and right by the linespacing size. It will flow from page to page until the default bodytext is reset by QQ. Like this:
- QQ**

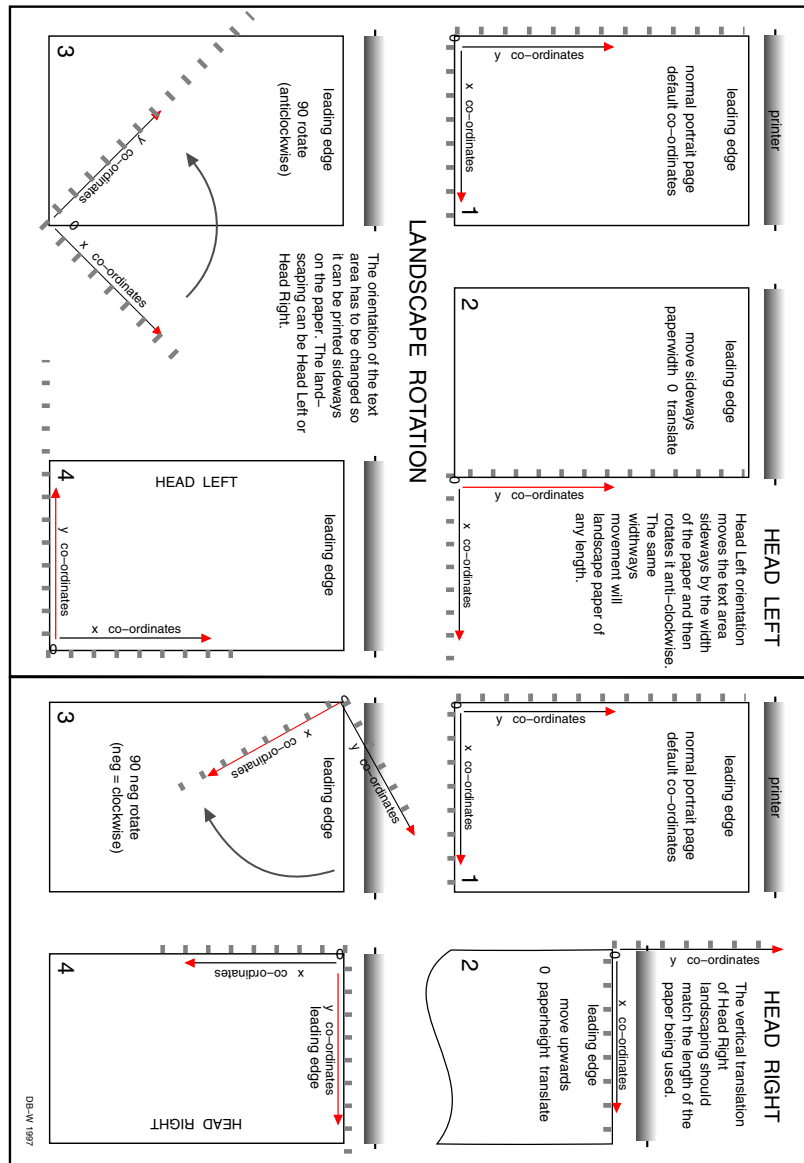
Inset and outset and other on-the-fly instructions are typed in upper case so that they may be more easily found in the script.

## % 7. TEXT ORIENTATION %

**/swap { pw ph 2 div PW PH } def**  
**/headright { 0 ph translate 90 neg rotate swap } bind def**  
**/headleft { pw 0 translate 90 rotate swap } bind def**  
**/changeround { pw ph PW PH 90 rotate 0 ph neg translate } bind def**  
**/quarto { pw 2 div PW ph 2 div PH } def**

Headright and headleft landscape the text area over the paper and swap exchanges the paper height for the width and *vice versa* to do so. The headright orientation puts the zero co-ordinates at the leading edge of the paper for short edge feeding A4 laser printers or inkjets. Use headleft for long edge feeding tabloid machines so that the paper enters the printer the right way up. Changeround landscapes a sheet for printing brochures and quarto divides the paper for 4up work and turn or nested sections.





## % 8. TILING %

```

/front { /tiledown { doprint } def /tileup { noprint } def } def % first pass
/back { /tiledown { noprint } def /tileup { doprint } def } def % second pass
/doprint { 0 0 translate } def /noprint { 0 ph translate } def

```

% tiling for 4 x 4up nested pages: font/back for 1st sheet: inner/outer for 2nd %

```

/side { pw 2 mul neg 0 translate } def
/inner { /tiledown { side doprint } def /tileup { side noprint } def } def
/outer { /tiledown { side noprint } def /tileup { side doprint } def } def

```

Type 'front' or 'back' at the head of the script for two sided printing for 2up folio binding or 4upA4 work and turn. Check 'back' with a % to print the front pages (see page 7). It is not needed for 1up or any 2up commands. If you type 'front 4upA4' and then flip the paper sideways for 'back 4upA4', you get two identical side by side copies in folio binding sequence.

Nested pages for sewn or stapled sections on tabloid A3 sheets use the same 'front' and 'back' sequence for the first sheet, and then 'inner' and 'outer' for the last. Replace all the % checks before using front again.

## % 9. SINGLE PAGE FORMATS %

Copy any of the following formats into the setup of a typesetting file. Insert your own running text and alter the numbering style by inserting 'headers', or 'number folio' in place of 'footers' or leave it empty. The 'number folio' places an incrementing number at the bottom of the page. To change numbering from headers to a folio for a chapter heading see section 21.

Changes to the page defaults needed for alternative paper sizes, such as the RM, FM, or IM values, should be inserted after the opening paper size. Substitutions for the textbox margin defaults for TM and RM must be typed *after* the resetting the margin in the textbox. North American users should substitute LETTER and TABLOID in place of A4 and A3.

```

/1upA5 { _Z A5 % level 1 should substitute 'a5 420 PW 595 PH' for A5
tinydict begin /textbox { margins bodytext } def
/chapter (Cappella Archive) def /title (Tiny Markup ) def
/numbering { footers } def p1 } def

```

The 1upA5 format sets single book size pages for on-screen proofing or re-imposition into sections or printer's pairs.



```

/1upA4 { % single pages for letters or reports
_Z A4 480 RM tinydict begin
/a4margins { margins 700 TM 480 RM } def % default margins adjusted for a4
/textbox { a4margins bodytext } def
/chapter (Cappella Archive) def /title (Tiny Markup ) def
/numbering { number folio } def p1 } def % noheaders or footers

/1upA3 { % single pages for posters: no default numbering
_Z A3 740 RM 72 FM 42 IM tinydict begin
/a3margins { margins 1030 TM 740 RM } def
/textbox { a3margins bodytext } def
/chapter (Cappella Archive) def /title (Tiny Markup ) def
p1 } def

```

## % 10. TWO-UP PAGE FORMATS %

```

/2upA4 { % four page folios
_Z A4 headright tinydict begin
/textbox { margins bodytext } def
/chapter (Cappella Archive) def /title (Tiny Markup ) def
/numbering { footers } def 4pages p1 } def

```

The automatic text flow provides a sequence of four library size book pages for printing in folding order 4–1: 2–3. Use ‘front 2upA4’ for the first pass and ‘back 2upA4’ for the reverse. Adjust the PH value slightly for more precise horizontal front/back register. There must be no space in ‘4pages’.

```

/2upPP { % printer's pairs for booklets or multi-page sections
_Z A4 headright tinydict begin
/textbox { margins bodytext } def
/chapter (Cappella Archive) def /title (Tiny Markup ) def
/numbering { footers } def 2pages p1 } def

```

These are pages paired in the correct sequence for stapled booklets. Proof the copy first as 1upA5 and save each page as a separate file. If you have to split the text to do so, use ) fj to fully justify the last line of the previous page and ( to prefix the text on the next. Remember to leave a space before the fj bracket.

After proofing, insert the following file header in each one:

```

%%Page: # # % section and page no.: e.g. 1 1 . . . 1 2, etc
16 PG % footer or header page number
newpage % the first pages 1 and 15 open with 2upPP

```

Copy the separate page files into front and back pairings, making sure the odd pages come first as ‘p1’ so as to be printed as the right hand (recto) of the facing page pair: e.g.

**% front: 1/16 : 3/14 : 5/12 : 7/10 back: 15/2 : 13/4 : 11/6 : 9/8**

Paste ‘2upPP’ at the head of the file and ‘close’ at the end. Print the front file first and then the second on the reverse.

```

1 PG 2upPP [return] (filepath:01page.ps) useDistiller
16 PG newpage [return] (filepath:16page.ps) useDistiller

```

You could also make a control file for the back and front shuffled pages, introducing each page file, followed by a Distiller importing filepath. See section 15.

```

/2upLR { % facing pages for left -> right on-screen reading
_Z A4 headright tinydict begin
/textbox { margins bodytext } def
/chapter (Cappella Archive) def /title (Tiny Markup ) def
/numbering { footers } def 2screen p1 } def

```

This format creates two facing pages for left to right on-screen reading as a PDF document.

## % 11. FOUR PAGE FORMATS %

```

/4upA4 { % work and turn sideways: toggle 'front/back 4upA4'
_Z A4 swap 0.71 dup scale tinydict begin
/textbox { margins bodytext } def
/chapter (Cappella Archive) def /title (Tiny Markup ) def
/numbering { footers } def 4pages p1 } def

```

A ‘front 4upA4’ at the head of script will print the first pass and an unchecked ‘back’ prints the reverse. Flipped sideways this produces two miniature copies for pagination proofs.

```

/4upA3 { % work and turn sideways: toggle 'front/back 4upA3'
_Z A3 quarto tinydict begin
/textbox { margins bodytext } def
/chapter (Cappella Archive) def /title (Tiny Markup ) def
/numbering { footers } def 4pages p1 } def

```

4upA3 work and turn produces full size 4up library book pages. Toggle ‘front 4upA3’ and ‘back 4upA3’. Use the 8 or 16 pages pagination for sewn or stapled sections.



**% 12 EIGHT PAGE FORMATS %**

```
/8booklet { % eight page folding: toggle 'front/back 8booklet'
_Z A3 quarto tinydict begin
/textbox { margins bodytext } def
/chapter (Cappella Archive) def /title (Tiny Markup ) def
/numbering { footers } def 8pages p1 } def
```

The eight page format is suitable for stapled instruction manuals from single sheets of A3 or tabloid paper. Use the front/back toggle.

```
/8wide { % eight pages landscaped: toggle 'front/back 8wide'
_Z A3 500 RM 60 FM 42 IM
headright ph 2 div PH tinydict begin
/8widemargins { margins 300 TM 460 RM } def
/textbox { 8widemargins bodytext } def
8card p1 } def
```

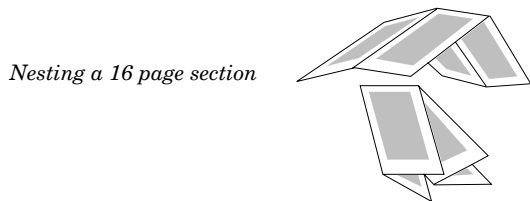
An eight page folding for anniversary cards for binding short or long edge. See page 41 for 8card pagination.

**% 13. SIXTEEN PAGE FORMATS %**

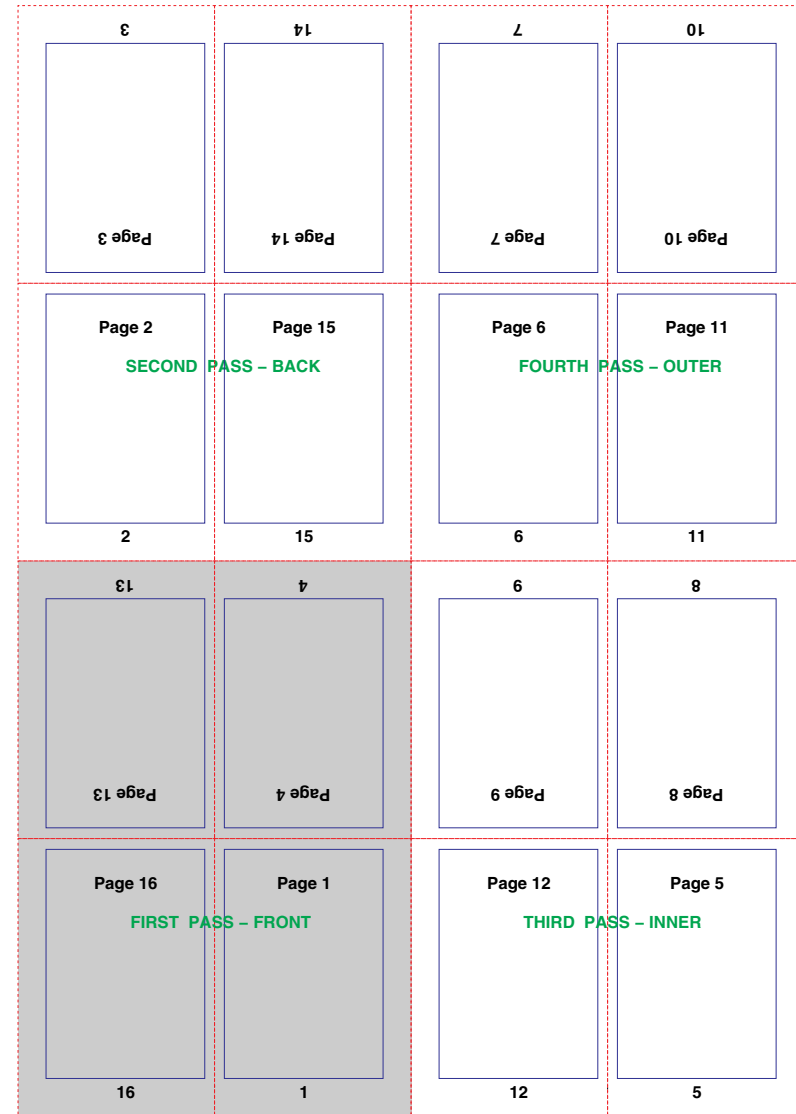
```
/16legal { % nested legal: toggle 'front/back: inner/outer 16legal'
_Z LEGAL 230 RM 60 FM 30 IM quarto tinydict begin
/textbox { margins 370 TM 230 RM 10 LG 8.5 rom MT } def
/chapter (Cappella Archive) def /title (Tiny Markup ) def
/numbering { footers } def 16pages p1 } def

/16nested { % nested A3: toggle 'front/back: inner/outer 16nested'
_Z A3 quarto tinydict begin
/textbox { margins bodytext } def
/chapter (Cappella Archive) def /title (Tiny Markup ) def
/numbering { footers } def 16pages p1 } def
```

The nested section is a 16 page imposition whereby two sheets of paper are printed 4up on either side and folded for the pages to appear in sequence. The result may be stapled as a booklet or several sewn together as sections for longer books.



Here are pages as they are tiled. Prefix '16nested' with front, back, inner, and outer, to switch the four sides in turn to the shaded printing area.



```

/tinytile { % tinytiles pagination proofs: US use: LETTER . . . 0.32 dup
_Z A4 300 RM 72 FM 42 IM
swap 0.35 dup scale tinydict begin
/textbox { margins bodytext } def
/chapter (Cappella Archive) def /title (Tiny Markup ) def
/numbering { footers } def 16consec p1 } def

```

TinyTile places sixteen consecutive miniature pages on one sheet of paper as pagination proofs. This provides a convenient method of checking page jumps, widows and orphans, and chapter headings without wasting paper or toner.

#### % 14. COLUMN FORMATS %

Columns may be opened on any page format by typing ‘3 NC 1st’ and the text will flow in three columns from one to the next and page to page until the normal bodytext is restored by using ‘nocols’. The default number of columns is two and the maximum five. A news-sheet may be created using 1upA3 4 NC 1st.

```

/brochure { _Z A4 setpaper changeround 30 FM 30 OM
pw om 2 mul sub RM tinydict begin
/coltext { 12 LG 10 rom MT } bind def
/textbox { margins 520 TM pw om 2 mul sub RM bodytext } def
p1 1st } def

```

A landscaped page for multi-column brochures. The text is landscaped on the paper and the footer margin reduced. The width of the outer margins and the text area remaining is determined by OM. Open the format with the required number of columns, such as ‘3 NC brochure’.

```

/landscaped { _Z A5 setpaper changeround 40 FM 72 OM
pw om 2 mul sub RM tinydict begin
/coltext { 12 LG 10 rom MT } bind def
/textbox { margins 300 TM pw om 2 mul sub RM bodytext } def
p1 1 NC 1st } def

```

A landscaped single page is useful for placing colour plates sideways in the bookpages. Alternatively, the ‘rotatepict’ instruction may be used. See the following section.

#### % 15. IMPORTING EPS FILES % Distiller Placing Procedures

```

/newsize { SC sc mul URY sc mul URX } def % resize as required
/middle { rm lm add urx sub 2 div LM } def % centre image
/room? { ury tm gt { newpage middle } { middle } ifelse } def % enough room?
/down { tm ury sub TM } def % move down image height
/URX { /urx exch def } def 0 URX % BoundingBox width: zero default
/URY { /ury exch def } def 0 URY % BoundingBox height: zero default
/SC { /sc exch def } def 1 SC % scaling: 1=100 percent default
/SS { currentscreen 3 -1 roll pop setscreen } bind def % halftone value in ‘place’
/place { newsize middle down _Z lm tm translate sc dup scale
106 SS import } bind def % width: height: scale : place
/rotatepict { gs 90 pivot exch dup tm exch sub 2 div 0 tr exch place gr } bind def
/text { ZZ 0 LM MT L } bind def % restore the text

```

The place instruction sets the halftone screen and calls the Distiller RunFile procedure. If the image is too large for the available space, it can jump to a newpage if ‘room?’ is substituted for ‘middle’. The halftone value is inserted in the place definition but this may be over-ridden by a different one in the imported EPS file. Place the files and Tinydict in the same folder and avoid adding spaces to the filepath, especially at the end.

Use ‘rotatepict’ instead of ‘place’ to landscape a graphic that is too wide for the measure. The ‘text’ command restores the normal typesetting co-ordinates just below the graphic. See section 20 for placing side by side images or runarounds in columns.

To import an EPS file with a known size use the ‘place’ instruction after the filepath, inserting the image Bounding Box height, width, and a scaling value, where 1 = 100%. If in doubt, guesstimate the dimensions and measure after proofing. Here are some filepath examples:

```

Macintosh (MacintoshHD:folder:filename.eps) 50 100 0.8 place [return] text
Windows (c:/mydir/filename.eps) 50 100 0.8 place[return] text
Unix (/mydir/filename.eps) 50 100 0.8 place [return] text

```

You may use Distiller to import an EPS file whose size is NOT known by using the following syntax which puts the EPS at the bottom of the page where it may be measured. Previously coded Tinydict files are imported in the same way and the ‘text’ replacement is NOT needed for either.





Macintosh (MacintoshHD:folder:filename.eps) useDistiller [return]  
Windows (c:/mydir/filename.eps) useDistiller [return]  
Unix (./mydir/filename.eps) useDistiller [return]

### Using a Document Manager

On a Macintosh you may let a document manager find the file but the filepath has to come *after* the ‘place’ command, as Adobe uses two different methods of quoting filepaths; one for Distiller and another for the PostScript Include Resource facility. If both were used in a file, it would look like this:

```
(MacintoshHD:PDFfolder:mylogo.eps)          % Distiller filepath
100 200 1 place [return]                      % width, height, scale
%%IncludeResource:mylogo.eps [return]        % DSC filepath
text [return]                                % restore bodytext
```

The different importing filepaths may be resolved by using the following toggle which searches for Distiller and picks up the alternative if it is not found. Any Distiller importing filepath is avoided when the same file is downloaded to a printer.

```
/noDistiller { userdict begin /import { clear } def end } def % laser printing only
/useDistiller {userdict begin /import { dup = flush RunFile } def end } def
/popDistiller { userdict begin /import { pop } def end } def
systemdict product (Distiller) search
{ pop useDistiller }{ pop pop noDistiller } ifelse
```

If the file only has Include Resource instructions or contains a file-merged or manually pasted EPSF after the ‘place’ command, type ‘noDistiller’ in the file Setup. If you need to insert a Distiller filepath in a file whose other images are file-merged head the file with ‘noDistiller’; prefix the relevant imported image with ‘useDistiller’ and reset ‘noDistiller’ afterwards!

### Manual Merging

Open a copy of the marked-up file in a long text editor and divide the file into sections corresponding with the appearance of each EPS image. Type ‘noDistiller’ at the head of the script and paste or file-merge the EPS file at the end after the ‘place’ command. Restore the bodytext by typing ‘text’ at the end of the EPS file. Increase the memory for the text editor if the graphic is a large one.

```
end % close the userdict opened in section 1.
%%EndSetup
```

## % 16. THE TINYDICT %

```
%%BeginDictionary
/tinydict 160 dict def % increase memory value for a level 1 ‘dictfull’ error.
tinydict begin % open the dictionary.
% hyphenation controls
/hyphenson {/split { splitword } def } bind def
/hyphensoff {/split {} def } bind def
tinydict /splitword known not { hyphenson } { hyphensoff } ifelse
```

## % 17. ABBREVIATIONS %

% Do not use abbreviations in sections 1 to 15 of this Resource. %

```
/ld { load def } def /rt /rmoveto ld /st /stroke ld
/bd { bind def } def /li /lineto ld /np /newpath ld
/gs /gsave ld /rl /rlineto ld /cp /closepath ld
/gr /grestore ld /ct /curveto ld /s /show load def
/mt /moveto ld /ro /rotate ld /w /widthshow load def
/tr /translate ld /rpt /repeat ld
/set { gs setlinewidth st gr } bind def
```

The Tinydict uses very few abbreviations so that definitions remain reasonably intelligible. The ‘load’ transfers system commands, whilst ‘bind’ marries a group of procedures for faster recall. The gsave and grestore twins ‘gs’ and ‘gr’ travel in pairs to isolate changes in colour, rotation, translation, and scaling from subsequent events. Use ‘widthshow’ to adjust individual lines of text by stretching or reducing spaces in conjunction with the spacewidth code ‘sw’. The ‘set’ is a variable that sets the width of a drawn line and uses a ‘gsave – grestore’ pair to preserve the co-ordinates for re-use.

## % 18. PAGE LAYOUT %

```
/recto { im fm tr textbox numbering } def % right hand page
/verso { pw rm im add sub 2 add fm tr textbox numbering } def % left hand page
/midpage { pw rm sub 2 div fm tr textbox numbering } def % centred page
```

The recto and verso position the facing page textboxes and midpage centres one on a single page. Choose running headers, footers, or footer folios by inserting your choice in ‘numbering’ in the page format. To outline a page, insert ‘outline’ in the textbox before ‘margins’.





**% 19. PAGINATION %**

```
/p1 { gs midpage % single pages for proofing or PDF display
/jump { bm tm gt { showpage gr p1 } if } def
} def

/2pages { % odd pages right handed (recto)
/p1 { gs pw 0 tr recto /jump { bm tm gt { gr p2 } if } def } def
/p2 { gs 0 0 tr verso /jump { bm tm gt { showpage gr p1 } if } def } def
} bind def

/2screen { % odd pages left handed (verso)
/p1 { gs 0 0 tr verso /jump { bm tm gt { gr p2 } if } def } def
/p2 { gs pw 0 tr recto /jump { bm tm gt { showpage gr p1 } if } def } def
} bind def

/4pages { % toggle front/back at head of script
/p1 { gs tiledown pw 0 tr recto /jump { bm tm gt { gr p2 } if } def } def
/p2 { gs tileup 0 0 tr verso /jump { bm tm gt { gr p3 } if } def } def
/p3 { gs tileup pw 0 tr recto /jump { bm tm gt { gr p4 } if } def } def
/p4 { gs tiledown 0 0 tr spinemark verso
/jump { bm tm gt { showpage gr p1 } if } def } def
} bind def

/8pages { % eight pages: one sheet
/noprint { 0 ph 2 mul tr } def /turn { 0 neg 0 tr 180 ro } def
/p1 { gs tiledown pw 0 tr recto
/jump { bm tm gt { gr p2 } if } def } def
/p2 { gs tileup 0 0 tr verso
/jump { bm tm gt { gr p3 } if } def } def
/p3 { gs tileup pw ph 2 mul tr turn recto
/jump { bm tm gt { gr p4 } if } def } def
/p4 { gs tiledown pw 2 mul ph 2 mul tr turn verso
/jump { bm tm gt { gr p5 } if } def } def
/p5 { gs tiledown pw ph 2 mul tr turn recto
/jump { bm tm gt { gr p6 } if } def } def
/p6 { gs tileup pw 2 mul ph 2 mul tr turn verso
/jump { bm tm gt { gr p7 } if } def } def
/p7 { gs tileup pw 0 tr recto
/jump { bm tm gt { gr p8 } if } def } def
/p8 { gs tiledown 0 0 tr verso
/jump { bm tm gt { showpage gr p1 } if } def } def
} bind def
```

```
/16pages { % sixteen pages nested: two sheets
/turn { 0 neg 0 tr 180 rotate } def
/p1 { gs tiledown pw 0 tr recto
/jump {bm tm gt { gr p2 } if } def }def
/p2 { gs tileup 0 0 tr verso
/jump {bm tm gt { gr p3 }if } def } def
/p3 { gs tileup pw ph 2 mul tr turn recto
/jump {bm tm gt { gr p4 } if } def } def
/p4 { gs tiledown pw 2 mul ph 2 mul tr turn verso
/jump {bm tm gt { gr p5 } if } def } def
/p5 { gs tiledown pw 3 mul 0 tr recto
/jump {bm tm gt { gr p6 } if } def } def
/p6 { gs tileup pw 2 mul 0 tr verso
/jump {bm tm gt { gr p7 } if } def } def
/p7 { gs tileup pw 3 mul ph 2 mul tr turn recto
/jump {bm tm gt { gr p8 } if } def } def
/p8 { gs tiledown pw 4 mul ph 2 mul tr turn verso
/jump {bm tm gt { gr p9 } if } def } def
/p9 { gs tiledown pw 3 mul ph 2 mul tr turn recto
/jump {bm tm gt { gr p10 } if } def } def
/p10 { gs tileup pw 4 mul ph 2 mul tr turn verso
/jump {bm tm gt { gr p11 } if } def } def
/p11 { gs tileup pw 3 mul 0 tr recto
/jump {bm tm gt { gr p12 } if } def } def
/p12 { gs tiledown pw 2 mul 0 tr spinemark verso
/jump {bm tm gt { gr p13 } if } def } def
/p13 { gs tiledown pw ph 2 mul tr spinemark turn recto
/jump {bm tm gt { gr p14 } if } def } def
/p14 { gs tileup pw 2 mul ph 2 mul tr turn verso
/jump {bm tm gt { gr p15 } if } def } def
/p15 { gs tileup pw 0 tr recto
/jump {bm tm gt { gr p16 } if } def } def
/p16 { gs tiledown 0 0 tr spinemark verso
/jump { bm tm gt { showpage gr p1 } if } def } def
} bind def
```



```

/16consec { % sixteen consecutive pages: one sheet
/p1 { gs 0 ph 3 mul translate verso
  /jump { bm tm gt { p2 } if } def } def
/p2 { gr gs pw ph 3 mul translate recto
  /jump { bm tm gt { p3 } if } def } def
/p3 { gr gs pw 2 mul ph 3 mul translate verso
  /jump { bm tm gt { p4 } if } def } def
/p4 { gr gs pw 3 mul ph 3 mul translate recto
  /jump { bm tm gt { p5 } if } def } def
/p5 { gr gs 0 ph 2 mul translate verso
  /jump { bm tm gt { p6 } if } def } def
/p6 { gr gs pw ph 2 mul translate recto
  /jump { bm tm gt { p7 } if } def } def
/p7 { gr gs pw 2 mul ph 2 mul translate verso
  /jump { bm tm gt { p8 } if } def } def
/p8 { gr gs pw 3 mul ph 2 mul translate recto
  /jump { bm tm gt { p9 } if } def } def
/p9 { gr gs 0 ph translate verso
  /jump { bm tm gt { p10 } if } def } def
/p10 { gr gs pw ph translate recto
  /jump { bm tm gt { p11 } if } def } def
/p11 { gr gs pw 2 mul ph translate verso
  /jump { bm tm gt { p12 } if } def } def
/p12 { gr gs pw 3 mul ph translate recto
  /jump { bm tm gt { p13 } if } def } def
/p13 { gr gs 0 0 translate verso
  /jump { bm tm gt { p14 } if } def } def
/p14 { gr gs pw 0 translate recto
  /jump { bm tm gt { p15 } if } def } def
/p15 { gr gs pw 2 mul 0 translate verso
  /jump { bm tm gt { p16 } if } def } def
/p16 { gr gs pw 3 mul 0 translate recto
  /jump { bm tm gt { showpage grestore p1 } if } def } def
} bind def

```

## % 20. COLUMNS %

```

/colheight { ph fm 2 mul sub TM } def % brochure column height
/colwidth { rm nc div gutter sub RM } def % based on value of NC
/gutter { lg 2 div } def % half linespacing
/match { tm vs sub tm exch sub TM } def % aligns next column with 'VS'
/nocols { tm end gr textbox TM H } def % 'end' removes the column dictionary
/1st { gs colwidth coltext MT VS
  /coldict 10 dict def coldict begin
  /jump { bm tm gt { 2 nc gt { nocols newpage 1st R }
    { 2nd } ifelse } if } def } bind def
/2nd { gr gs rm lg add 0 tr coltext vs TM MT
  /jump { bm tm gt { 3 nc gt { nocols newpage 1st R }
    { 3rd } ifelse } if } def } bind def
/3rd { gr gs rm lg add 2 mul 0 tr coltext vs TM MT
  /jump { bm tm gt { 4 nc gt { nocols newpage 1st R }
    { 4th } ifelse } if } def } bind def
/4th { gr gs rm lg add 3 mul 0 tr coltext vs TM MT
  /jump { bm tm gt { 5 nc gt { nocols newpage 1st R }
    { 5th } ifelse } if } def } bind def
/5th { gr gs rm lg add 4 mul 0 tr coltext vs TM MT
  /jump { bm tm gt { nocols newpage 1st R } if } def } bind def

```

This revised mark-up creates up to five columns across a page. The number of columns is chosen by the value of NC, two being the default. The font size, style and linespacing of your choice are inserted in the coltext instruction. The VS vertical store recalls the height on the page for placing vertical rules or opening another column at the same or other chosen height.

### 1st

Always open the 1st column, even if it is empty, and then chose the others in any order. To return to the 1st from one of the others, type 'nocols 1st' or it will be misplaced. The gutter between the columns is the same as the linespacing LG.

**L % add a linespace to level columns**

### 2nd % open the 2nd column.

Text will automatically jump from column to column and page to page. A forced jump is made by typing the column number and a forced page jump by specifying 'nocols newpage 1st'. Close the columns with 'nocols' before changing the value of NC.

**LR % insert a left vertical rule  
nocols % close columns**



Type ‘nocols’ to close the last column, to move back to the first column, or to change the number, such as the ‘nocols 3 NC 1st’. A newpage is forced by ‘nocols newpage’, if a nocols is missing, the following columns will not be in the correct place. You may open the columns at a specific page position such as ‘100 TM MT 1st’.

**3 NC 1st** % ‘nocols’ not needed to open columns  
Alter NC to change the number.

**2nd match** The ‘match’ places the first line level with VS in the previous column.

**3rd** The sequence here is 1st, 3rd, 2nd, with VS placed in column 3 at the point where the 2nd is to open.

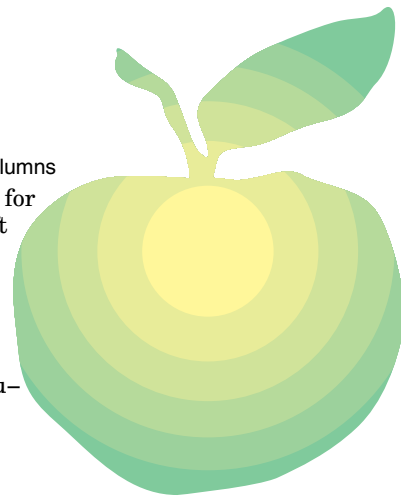
**VS** If you use VS more than once per column, only the last one is recognized.

**nocols 1st** % reopen 1st column

To switch back to a first column from one of the others, type ‘nocols 1st’.

**nocols 2 NC 1st** % change from 3 to 2 columns

The 1st and 2nd columns are useful for placing a graphic to the right or left on the page instead of the default position in the centre. The accompanying text may be placed to one side or the other and after proofing each line may be adjusted with an ‘sw’ coding to wrap around an irregular shape as happens here.



**2nd (dir:folder:apple.eps)**

**100 120 1 place text nocols 1st**

% restore bodytext with ‘text nocols’.

For a runaround stretch the spaces in the shorter lines with a points value in front of the space width ‘sw’ and ending with an abbreviated ‘widthshow’ and a line advance. Use ‘P’ to justify as here; proof the page and divide the end of line white space by the number of spaces in the line. Try to guesstimate the value by eye!

**2nd**

(The 1st and 2nd columns are useful for) n  
0.5 sw (placing a graphic to the right or left) w L  
1.2 sw (on the page instead of the default) w L  
1 sw (position in the centre. The accom–) w L  
0.5 sw (panying text may be placed to one) w L  
(side or the other and after proofing) n  
1 sw (each line may be adjusted with an ) w L  
(‘sw’ coding to wrap around an irregu–) n  
(lar shape as happens here. ) n

## % 21. RUNNING TEXT %

**/chapter (Cappella Archive) def**

**/title (Tinydict Markup) def**

Insert your own running text for headers and footers and place it in the page format in the file header. On the printed page the title appears right handed (recto) and the chapter left handed (verso). Change the chapter text before forcing a newpage and redefine numbering to place a folio footer page number. Restore the running headers for the following pages of the chapter. Like this:

**... last line of text. ) J**

% end of previous chapter

**/chapter (Chapter Two) def**

% next chapter header

**/numbering { number folio } def** % remove headers: number next page as a folio  
**newpage** % force a new page

**/numbering { headers } def** % restore headers for following pages

**40 a** % drop the chapter header down the page: your distance in points

**15 it chapter c L 10 rom** % 15 pt italics: centre text: advance: 10 pt roman

**(This is the first line of the next chapter ...**

If there is no running header or footer text, leave the chapter heading empty, and the sixth line could read:

**15 it (School Days) c L 10 rom**

## % 22. HEADERS %

**/headers { % 2 b 0.2 rule H**

% uncheck % for a header rule

**/numodd { gs 0 T 10 it chapter centre s 10 rom number**

**right s gr L H 2 a } def**

**/numeven { gs 0 T 10 rom number s 10 it title centre s gr L H 2 a } def**

**pg 2 mod 1 eq { numodd } { numeven } ifelse**

**} bind def**

The page numbering ensures the correct recto and verso placing of headers; odd numbers being placed ‘recto’ on the right hand page, and even numbers ‘verso’ to the left. The ‘0 T’ makes sure that headers are unaffected by any indented text styles. Change the font size and style to suit and insert ‘footers’ or ‘headers’ in ‘numbering’. Leave ‘numbering’ empty for preliminary pages and insert roman numbers, e.g., ‘(viii) folio’ after ‘newpage’.



## % 23. FOOTERS %

```
/footers {  
/numodd { gs 0 T 10 rom number centre footerproc 10 it chapter right  
footerproc gr } def  
/numeven { gs 0 T 10 it title footerproc 10 rom number centre footerproc gr } def  
pg 2 mod 1 eq { numodd } { numeven } ifelse  
} bind def
```

The running text is transferred to the footer and the page number centered. If the definition in ‘numbering’ is {number folio} the page number will be centered in the footer margin without any text.

## % 24. PAGE NUMBERING %

```
/footerproc { currentpoint pop bm 20 neg add moveto show } def  
/number { pg pg 1 add PG 4 string cvs } def  
/numbering { } def  
/folio { gs pg pop 50 string cvs centre footerproc gr } def
```

Specify other starting numbers when required, such as ‘5 PG 2upPP’ or ‘16 PG newpage’. The ‘folio’ will centre any typed text at the foot of the page when ‘numbering’ in the page formats is left empty. After a newpage simply type ‘number folio’, (xxi) folio, (page two) folio: (Preface) folio: Always start with an odd page number.

See the Tiny Resources for section numbering procedures.

## % 25. HORIZONTAL MOVEMENT %

```
/emsize {lg 0.825 mul 0 rt } bind def % type ‘em’ before any text  
/em { bm tm gt { jump emsize } { emsize } ifelse } def % page jumps  
/en { lg 0.5 mul 0 rmoveto } bind def % half the linespacing  
/centre { dsp 2 div textwidth 2 div exch sub lm add tm moveto } bind def  
/gutter { lg 2 div } def % gutter spacing  
/HS { currentpoint pop /hs exch def } def % store horizontal position  
/k { 0 rmoveto } def % move horizontally: minus left, plus right: e.g. -1.5 k  
/right { dsp rm exch sub tm moveto } bind def % see section 31 for ‘dsp’  
/INSET { lm in add LM rm in sub RM MT } bind def % inset by value of IN  
/OUTSET { lm in sub LM rm in add RM MT } bind def % outset by value of IN  
/TI { lm gutter add LM rm gutter sub 2 add RM MT } def % in half linespacing  
/TO { lm gutter sub LM rm gutter add 2 sub RM MT } def % out half linespacing
```

You may move horizontally one em for a first line paragraph indent by placing it individually, using the ‘M’ code, or adding it to ‘P’ in

% 30 %

% *The Tiny Guide*

section 34. The en space is useful for equal spacing of figures which are normally one en wide. The text may be shifted to the right or the centre and these instructions are part of the equivalent mark-up code. Type them out in full when using the colour code CS, such as **centre red CS L**

Text may be indented by half the linespacing using TI and TO and this occurs automatically when a line box LB is placed, but not a filled box. For inset text over a page jump use an alternative text style, such as the quotetext suggested in section 6.

## % 26. VERTICAL MOVEMENT %

```
/MT { 0.1 a } bind def % forwards 0.1 point: used for new margin values  
/a { tm exch sub TM lm tm mt } bind def % points advance: e.g. 100 a  
/A { lg 2 div a } bind def % forwards half a line  
/b { tm add TM lm tm mt } bind def % points backwards: e.g. 20 b  
/B { lg 2 div b } bind def % backwards half a line  
/L { tm lg sub TM lm tm moveto } bind def % forwards one line  
/R { tm lg add TM lm tm moveto } bind def % reverse one line  
/newpage { 10 neg TM tm pop jump } def % force a page break  
/pivot { currentpoint exch translate rotate } def % rotate on the spot  
/inf { gs 0 lg 4 div neg rt 0.7 dup scale s gr lg 4 div 0 rt } bind def % inferior  
/sup { gs 0 lg 4 div rt 0.7 dup scale s gr lg 4 div 0 rt } bind def % superior  
/v { 0 exch rmoveto } def % move vertically: minus up, plus down: e.g. 2 v  
/VS { tm /vs exch def } def % store current vertical position
```

Superior and inferior figures are placed with ‘sup’ and ‘inf’ and are used for setting formulae or indicating footnotes. They are not genuine ones, because the current typeface is shrunk by 30%, whereas the real thing is better proportioned. However, it works for footnote<sup>1</sup> notation and for chemical formulae. For example:

**(H) s (2) inf (SO) s (4) inf** becomes H<sub>2</sub>SO<sub>4</sub>

The ‘a’ or ‘b’ can move any vertical distance great or small: e.g. 120 a or even 0.015 b. The pivot instruction rotates drawings or text on the spot, whereas the normal PostScript rotate revolves in relation to the lower left hand corner of the page. Use ‘neg’ to go clockwise and isolate the rotation with a gs/gr pair.

**gs (Pivoted text) 15 neg s gr** *Pivoted text*

---

1. Like this one.

% *PostScript Markup*

% 31 %



90 IN 1 yellow FB

90 IN B SB

% 27. BOXES %

1 blue LB

```

/boxdrop { lg mul lg 2 div add } bind def
/ypos { vs tm sub } def
/drawbox { 0 ypos rl rm lm sub 0 rl 0 ypos neg rl closepath } def
/SB { B VS INSET TI L } def
/LB { B TO gs cmyk drawbox setlinewidth stroke gr OUTSET 0 IN L } bind def
/CB { 0.2 black LB } def
/FB { B VS INSET gs cmyk boxdrop dup a drawbox fill b gr OUTSET 0 IN L } bind def
/FR { R VS _Z OUTSET 0 TM LB ZZ L } bind def
/frame { B urx rm exch sub 2 div IN INSET TI } def

```

SB

The box mark-up in the earlier versions of the Tinydict has been simplified. The line box LB outlines in any colour and has the sequence linewidth – colour – LB. The CB box is a shorthand version of LB and places a thin black outline round the text, as it does here. (Old letterpress compositors called it a ‘coffin’). Both LB and CB need the start box instruction SB and the enclosed text is inset all round by half the linespacing.

CB

7 reded FB TI

% line drop: colour: filled box: text inset

The filled box FB floods an area with colour and the mark-up is: drop – colour – FB. It must be placed before text, which would otherwise be obscured and the drop is in multiples of the linespacing. The bottom of the box fits between the lines of text. Guesstimate the drop and correct from a printed proof or PDF conversion. Line boxes indent the text automatically but filled boxes need TI text in and TO text out as shown here.

TO H

% text outset: H half line advance

3 NC 1st

The frame outlines imported eps graphics and here is placed round an image in the second of three columns. The VS vertical store is placed before the eps filepath is typed.

2nd VS (dir:folder:me.eps)



73 68 1 place text  
frame 1 red LB

3rd

After the ‘text’ return, the ‘frame’ is followed by the line width and colour and then by LB. Here is your author before PostScript turned his hair white! **nocols**

% 32 %

% The Tiny Guide

40 IN SB

% inset box by 40 points: start a line box

The width of a box is varied by the value of IN. If the number is minus, the box will expand beyond the textbox margins. The inset here is 40 points and the default margins and value of IN are always restored when a box is closed.

2 red LB

% close the box with a 2 point red outline.

% Text in a line box is automatically inset left and right.

50 IN 5 yellow FB

% 50 pt inset: 5 line drop: colour

B 50 IN SB

% back a half line: 50 pt inset: start a line box

A filled box may be outlined by placing an FB followed by an LB. Any inset value must be quoted for each and the start box SB must be moved up the page by a half line reversal B to level both boxes.

1.5 blue LB

Boxes may be used to make some titling suitable for colour printed posters or letterheads. Notice the duplication of the text to provide a shadow and use of gs/gr to superimpose the one over the other.

**CAPPELLA ARCHIVE**  
**PRACTICAL POSTSCRIPT**

4 palegreen FB 3 a 12 IN 2.5 paleyellow FB R 4 b SB

L H 24 rom gs 13 T ( CAPPELLA ARCHIVE ) dup 0.5 setgray s gr

1 a 11 T red CS L

13 bol ( PRACTICAL POSTSCRIPT ) centre blue CS L

L H 1 red LB

The instructions are: make a 4 line drop pale green filled box: advance 3 points and inset 12 points. Open a 2.5 line drop yellow box: reverse the distance travelled by one line and 4 points back up the page. Start a line box with SB and advance a line and a half with L H. Duplicate the text and place and colour one in 24 point roman in gray and the other in red and in 13 point bold: advance a line and a half and close the box with a 1 point red outline.

% PostScript Markup

% 33 %





20 a 150 T 40 yellow FC  
2 35 red LC



% 28. CIRCLES %

```
/placecircle { currentpoint 3 -1 roll newpath 0 360 arc } def
/LC { gs cmyk placecircle setlinewidth stroke gr } bind def
/FC { gs cmyk placecircle fill gr } bind def
```

Like the boxes, there are two types of circle, the filled FC and outline LC. The mark-up sequence is: radius – colour – FB and linewidth – radius – colour – LC. Their centres are placed horizontally by a tabbing value given to T and vertically by an advance equal to their radius. As with boxes, filled circles should be placed before any outlines and both will travel with any editing of the text.

A useful variation of the circle is the oval or ellipse as a background to a graphic. The best way is to measure the image area, which, in this example is 140 points wide and 260 high. It is placed on the page by dropping down half the height (130), which then becomes the radius.

To centre it on the page, use half the value of RM in front of the tab instruction T. This T must come after the advance or it will not move. The scaling ratio to draw the narrower width of the oval will be the width divided by height which, as you see, PostScript can calculate for you, but don't forget the vertical 1 of the scaling ratio. Place all the mark-up between the gsave/grestore twins, or the distortion will continue for ever. The oval is defined thus:

```
gs 130 a 150 T 140 260 div 1 scale 130 palegreen FC 2 gr
```

% advance to centre: tab across: x y scaling ratio: radius: colour: width

Here it is. You would make any shift in position by adjusting the drop or tabbing values.

```
/partcircle { currentpoint 3 -1 roll newpath 90 330 arc } def
/HC {gs 5 1 roll cmyk setlinewidth partcircle stroke gr } bind def
```

Using partcircle makes it possible to print interlinking circles by superimposing a portion of a circle over a previously placed circle, thus:



% 34 %

% The Tiny Guide

## 29. RULES

```
/HR { 0.2 rule } def % draw a horizontal rule 0.2 points thick
```

```
/rule { B gs lm tm mt textwidth 3 sub 0 rlineto set gr L } bind def
```

% right/left vertical rules: use VS at start: RR or LR at finish %

```
/vline { 0 vs tm sub lg 2 div add neg rl 0.2 set } def
```

```
/RR { gs rm 1 sub gutter add vs lg add mt vline gr } def % right hand rule
```

```
/LR { gs lm 1 sub gutter sub vs lg add mt vline gr } def % left hand rule
```

% folio binding rising spinemark

```
/spinemark { gs pw pg 50 add mt 0 -4 rl 0.3 set gr } bind def
```

VS

HR is the default horizontal rule; or specify the width: e.g. 1.5 rule. For vertical rules type VS at the starting point and RR or LR at finish. The spinemark is a rising indicator for folding. Use TI and TO to level vertical rules with the text margins. **LR RR**

Inset, outset and rule codes can make short end-of-chapter rules:

```
gs red cmyk 100 IN INSET 2 rule 3 b 0.5 rule OUTSET gr
```

```
/myrect { 4 -2 roll mt 1 index 0 rl 0 exch rl neg 0 rl cp 0.5 set } bind def
```

```
/outline { gs [ 6 ] 0 setdash 0 0 mt 0 0 pw ph myrect 0.1 set gr } bind def
```

To reveal the page outlines, insert 'outline' in the textbox before the 'margins' instruction.

## 30. WATERMARK AND BACKGROUND

```
/watermark { gs 0 0 mt 60 ro 36 rom (Cappella Archive) 0.8 setgray s gr } def
```

```
/background { gs paleyellow cmyk 0 0 pw ph myrect fill gr } def
```

The sequence in the textbox should be background – watermark – margins – bodytext

## 31. DIGITAL ASSISTANTS

```
/space ( ) def % define a space
```

```
/sw { 0 32 } def % stretch space widths using 'w': e.g. 1.2 sw (wider spaces) w
```

```
/dsp { dup stringwidth pop } def % how long is the chosen text?
```

```
/cpp { currentpoint pop } def % how far along the textwidth?
```

```
/textwidth { rm lm sub } def % usually equals RM value
```

```
/ssp { space search pop } def % look for spaces
```

```
/dsc { dup spacecount } def % count them
```

```
/rejoin { ssp exch glue } def % rejoin a word to its space
```

These abbreviations make many procedures less verbose.

% PostScript Markup

% 35 %





### 32. SPACECOUNTING AND LINEWRAPPING

```

/find { search { pop 3 -1 roll 1 add 3 1 roll } { pop exit } ifelse } def
/spacecount { 0 exch space { find } loop } def
/toofar? { rejoin dsp cpp add rm gt } def

```

Toofar? defines a space; finds each one in the text, compares its position with the line width specified by the RM margin value, and returns a true or false comment.

### 33. JUSTIFICATION PROCEDURES

```

/glue { 2 copy length exch length add string dup 4 2 roll 2 index 0 3 index
putinterval exch length exch putinterval } bind def      % join strings of text
/TXT { tinydict /txt 3 -1 roll put } def () TXT           % make a word store
/startpoint { rm currentpoint pop sub } bind def         % length of line left
/measure { dsp txt stringwidth pop add startpoint 2 add gt } bind def
/howlong? { rejoin measure } def                         % length of text so far
/join { txt exch glue TXT } def                          % add word to previous text
/jproc { dsp startpoint exch sub exch dsc } bind def     % count spaces
/popzero { dup 0 eq { pop } { div } ifelse } def         % ignore if no space on line
                                                    % stretch spaces to fill the linewidth
/justify { jproc 1 sub 3 2 roll exch popzero sw 4 3 roll w L } bind def
                                                    % justify measured text: open the word store: put overlong text in store
/nextline { txt justify () TXT join } bind def           % carry excess words over

```

As explained in *Practical PostScript*, the full justification process is ‘an ill-favoured thing, but mine own’ and this is what happens. It finds a space; glues it back onto the previous word, if any; measures both; checks against the textwidth; rejoins them to any previous words and spaces; stores the measured text in a digital cupboard; adjusts the width of the spaces if the end of the line has been reached; checks the line position in relation to the bottom margin; places the line, moving to a new page if necessary; repeats for the remaining lines; places the last line flush left, and finally, moves into position for the next paragraph.

A word store is made for the measured text by providing an empty text string TXT. The popzero catches a last line with only one word and no spaces. The glue procedure is an involved method of joining two lengths of text together so fully justified text takes twice as long to print as does text flush left.

### 34. MARK-UP CODES

```

/s /show load def                                     % place text: stay on line
/n { bm tm gt { jump s L } { s L } ifelse } bind def   % place text: advance a line
/r { right s L } bind def                             % place text right: advance a line: no page jump
/c { centre s L } bind def                            % centre text: advance a line: no page jump
/T { lm pop tm moveto } def                           % horizontal tabs: # T
/VT { gs hs exch moveto s gr } def                   % vertical tabs: HS # T
/S { dsc { jump toofar? { L jump s } { s } ifelse } repeat pop } bind def
/CS { gs cmyk S currentpoint grestore moveto } def    % place coloured text
/P { S L } def                                        % place word, line, or paragraph: advance a line: page jump
/p { dsc { toofar? { L s } { s } ifelse } repeat pop L } bind def % no page jump

```

The linewrapping codes are in upper case letters and the non-linewrapping ones in lower case. Use ‘S’ for font changes and leave a space before the CS, S, J, and P closing bracket: e.g. (text.) S. The little ‘s’ abbreviates the ‘show’ placing instruction and should be used to position tabbed text across the page when using ‘T’.

The ‘n’ code sets text flush left and doesn’t linewrap (which is useful for lines stretching over columns) but it does page jump, and ‘c’ and ‘r’ set centre and flush right respectively. To set all three on the same line, use ‘s’ for flush left text, ‘centre s’ and then ‘r’. The ‘CS’ sets coloured text and needs instructions typed in full, such as ‘centre green CS’. Add an ‘L’ for a line advance.

The upper case codes will automatically page jump when the text reaches the bottom margin. The last line on the page should use black text and the default font and size in the textbox or the word spacing on the next page may be peculiar. To avoid page jumping when setting footnotes or tables, use ‘s L’, ‘p’, or ‘nj’.

```

/j { dsc { howlong? { split jump nextline }
{ join } ifelse } rpt txt s () TXT pop } bind def
/nj { dsc { howlong? { split nextline }
{ join } ifelse } rpt txt s L () TXT pop } bind def
/fj { dsc { howlong? { split nextline }
{ join } ifelse } rpt txt justify () TXT pop } bind def
/J { dsc { howlong? { split jump nextline }
{ join } ifelse } rpt txt n () TXT pop } bind def
/M { em dsc { howlong? { split jump nextline }
{ join } ifelse } rpt txt n () TXT pop } bind def
/C { dsc { howlong? { jump txt c () TXT join }

```



```
{ join } ifelse } rpt txt c () TXT pop } bind def
```

The full justification codes need a space before the final bracket, or the last word will disappear into digital limbo. The ‘J’ code will justify a paragraph and little ‘j’ is used for in–line font changes or to avoid a linewidth . Use ‘j L’ to complete such a paragraph.

The ‘C’ centres an entire paragraph of text as here, but use little ‘c’ for single lines such as chapter headers. The ‘M’ definition will indent the first line of every paragraph. The ‘nj’ does not page jump and is used for setting footnotes whilst ‘fj’ force justifies a line or even single letters, like this:

**J U S T I F I C A T I O N**

```
/LS { dup length 1 sub 3 -1 roll dup 3 -1 roll mul 3 -1 roll dsp 3 -1 roll add
2 div textwidth 2 div exch sub lm add tm mt exch 0 3 -1 roll ashow L } bind def
```

Commercial typesetting software sometimes justifies text by not only adjusting the spaces but also the distance between each character. This form of letter spacing (or tracking as it is sometimes called) is a practice is frowned on by typographical purists but by tradition headings and titles are usually letter spaced. The LS instruction will do this and centre the heading on the page. The spacing distance in points is typed first followed by the text and then the mark–up. This example is **2 (LETTER SPACED) LS**

**NORMAL SPACING**  
**LETTER SPACED**

### 35. DEFINED COLOURS

```
/red { 0 1 1 0 } def /palered { 0 0.3 0.3 0 } def
/blue { 1 1 0 0 } def /paleblue { 0.3 0 0 0 } def
/green { 1 0 1 0 } def /palegreen { 0.3 0 0.3 0 } def
/yellow { 0 0 1 0 } def /paleyellow { 0 0 0.1 0 } def
/black { 0 0 0 1 } def /cmyk /setcmykcolor load def
```

PDF documents use rgb colours for on–screen display but cmyk provides a greater depth of printed colour on suitable inkjets. Add your own preferences above or below but do not give the same name to both rgb and cmyk versions.

```
/redrgb { 1 0 0 } def /greenrgb { 0 1 0 } def
/bluergb { 0 0 1 } def /rgb /setrgbcolor load def
```

### 36. DEFAULT TYPEFACES

```
/languagelevel 1 where {/F { findfont exch scalefont setfont } def }
{/F { exch selectfont } def } ifelse pop % enable level 1 interpreters
```

#### Default Typefaces

```
/rom { /Times–Roman F } def /sb { /Helvetica–Bold F } def
/bol { /Times–Bold F } def /si { /Helvetica–Oblique F } def
/it { /Times–Italic F } def /sbit { /Helvetica–BoldOblique F } def
/bolit { /Times–BoldItalic F } def /sym { /Symbol F } def
/ss { /Helvetica F } def /cr { /Courier F } def
```

These are the default typefaces for the bodytext. Substitute your own selection using the actual PostScript font name such as Times–Italic, not the operating system file name, (like TimesIta). Substitute another typeface by name or by opening another set, as discussed below. Any available PostScript Type 1 or 3 typeface may be used, but no TrueType fonts. Typing ‘10 rom’ will set the current roman typeface at 10 points; whilst ‘12 ss’ will provide a 12 point sans serif, and similarly ‘9 bol’ for bold, or ‘11 it’ for italic, and so on.

#### Grouped Typefaces

```
/GaraGillset {/rom { /Garamond–Roman F } def
/bol { /Garamond–Bold F } def
/it { /Garamond–Italic F } def
/bolit { /Garamond–BoldItalic F } def
/ss { /GillSans F } def
/sb { /GillSans–Bold F } def
/si { /GillSans–Italic F } def
/sbit { /GillSans–BoldItalic F } def
} def
```

The group name is put in the bodytext for a global font change and the new style pasted before a page jump. Otherwise type the name for that page only.

```
14 /GillSans–Bold F % this would specify a one–off use at 14 points.
(HEADLINE) c % centre the headline
GaraGillset 10 rom % restore body text size
```

The ‘10 rom’ would restyle the following text in 10 point Garamond Roman but the defaults will reappear on the next page unless the bodytext was redefined at this point: e.g.

```
/bodytext { 12 LG GaraGillset 10 rom MT } def
```



This new bodytext will continue on succeeding pages until it is changed again. Use the correct PostScript font names and one set, like the Times and Helvetica group above, must remain ungrouped to act as the default. Download non-printer fonts beforehand using the printer utility software.

Typefaces have different ‘stringlengths’ and the substitution of one for another may upset any hand hyphenation of lines or control of widows ‘n’ orphans. (Orphans are one or two words carried over to the top of a new page. Widows are paragraphs starting on the last line of a page.)

A few widows are inevitable with auto-flowed text and may be ignored unless spotted by Inspector Meticulous. You may control orphans by joining or splitting previous paragraphs, although this in itself may create new ones on following pages. Alternatively, look for paragraphs ending with a single word and amend the text to take it back.

## 37. RESOURCES

Copy and paste additional resource procedures into this section.

```
%/H { A } def % original advance code: uncheck to use
%/h { sw } def % original space width code: uncheck to use

% start footnotes: e.g. 41 FN
/FN { bm add BM _Z 8 rom 10 LG lm bm TM tm mt H HR } def
/EN {ZZ bodytext } def % end footnotes

% drop capitals: specify drop distance: character
% e.g. 12.4 LG (R) dc 10 rom 12 LG (est of text.)
/dc { findfont lm tm moveto [ lg 3.7 mul 0 0 lg 3.7 mul 0 lg 2 mul neg ]
makefont setfont 2 string cvs s currentpoint pop LM bodytext } def

% a fake bold: that looks like this
/bold { gs false charpath gs 0 setgray fill gr 1 setlinejoin 0.3 set gr } bind def

% indexing : (Here is some text ) s (text ) XX
/indexword { 8 rom show pg 1 sub 4 string cvs show } def
/rightmargin { gs rm 12 add tm moveto indexword gr } def
/leftmargin { gs -50 tm moveto indexword gr } def
/XX { pg 2 mod 1 eq { leftmargin } { rightmargin } ifelse } def

% underlining: e.g. (underline) ul
/ul { dup show stringwidth pop neg gsave 0 currentfont dup
/FontInfo get /UnderlineThickness get exch
/FontMatrix get dtransform setlinewidth 0 currentfont dup
/FontInfo get /UnderlinePosition get exch
/FontMatrix get dtransform rmoveto rlineto stroke
grestore } def

The TinyDivi hyphenation resource should be pasted here.
Substitute the TeX dividing patterns if preferred.

The dictionary entries must be closed by ‘end’ before it can be re-
opened in the page format as ‘tinydict begin’.
%%EndDictionary
end % the Tinydict must be closed here
%%EndResource
```



## ADDITIONAL RESOURCES

These resources are in this Guide:

PDF Resources	45
Font Re-encoding	49
TinyTables	54
Section Numbering	59
Setting Equations	60
Footnotes	63
Grep Mark-up Patterns	64

The following resources are also available from:

<http://www.cappella.demon.co.uk/tinyfiles/tinymenu.html>

Adding a euro to a font	T <sub>E</sub> X to 2up folio
Auto linespacing	TinyGraphs
BarCharts	1up PDF to 2up folios
Hyphenation test	TinyClef music printing
Mailing labels	

### Card Pagination

```
/8card { % use 'front 8wide': 'back 8wide'  
/noprint { 0 ph 2 mul tr } def /turn { 0 neg 0 tr 180 ro } def  
/p1 { gs tiledown pw 0 tr recto  
/jump { bm tm gt { gr p2 } if } def } def  
/p2 { gs tileup 0 0 tr verso  
/jump { bm tm gt { gr p3 } if } def } def  
/p3 { gs tileup pw 0 tr recto  
/jump { bm tm gt { gr p4 } if } def } def  
/p4 { gs tiledown 0 0 tr verso  
/jump { bm tm gt { gr p5 } if } def } def  
/p5 { gs tiledown pw ph 2 mul tr turn recto  
/jump { bm tm gt { gr p6 } if } def } def  
/p6 { gs tileup pw 2 mul ph 2 mul tr turn verso  
/jump { bm tm gt { gr p7 } if } def } def  
/p7 { gs tileup pw ph 2 mul tr turn recto  
/jump { bm tm gt { gr p8 } if } def } def  
/p8 { gs tiledown pw 2 mul ph 2 mul tr turn verso  
/jump { bm tm gt { showpage gr p1 } if } def } def  
} bind def
```

% 42 %

% *The Tiny Guide*

## PRINTING RESOURCES

% Remove the check % to use a procedure %

Laser printers increasingly rely on push-button menus and the more obvious PostScript level 1 control instructions like 'manual feed true' have been superseded. A selection follows which may be cut and pasted into the prolog. Uncheck those required. The manual feed (multipurpose tray) is recommended when printing, to keep an eye open for double feeds.

```
% serverdict begin 0 exitserver % level 1: do not use on a network.  
% true 0 startjob % level 2 onwards
```

Removing the % from 'exitserver' or 'startjob' puts the Tinydict into the PostScript printer, Distiller, or Ghostscript memory. Ignore the Distiller response of 'No PDF file produced'. Always use 'startjob' unless using a level 1 printer. Do NOT uncheck either of them if the Tinydict is pasted at the head of a marked-up file for printing, distilling, or archiving.

The following device-dependent printing instructions may be included at the head of a typeset file for laser-printing. Do not use any of them when distilling into PDF, printing on an inkjet, or in files transmitted for commercial printing.

**If they are downloaded to a printer with the 'exitserver' unchecked, they may alter the printer switch-on defaults until some revised commands are similarly sent.**

```
%/manual { statusdict begin /manualfeed true def end } def % Level 1  
% 0 = lowertray: 1 = multipurpose: % Level 2: check correct numbers from manual  
%/manual { statusdict begin mark 1 setpapertray cleartomark end } def  
%/normal { statusdict begin mark 0 setpapertray cleartomark end } def  
%/copies { userdict begin /#copies exch def end } def % e.g. 5 copies  
% A4 597 842 : letter 612 792 : legal 612 1008  
% B4 729 1032 : 11x17 792 1224 : A3 842 1190  
%<< /PageSize [ 597 842 ] >> setpagedevice % A4 sizes
```

Copy and paste the required instruction into the prolog and uncheck to use. When laser printing, do not vary the standard sizes by more than 5 points or an error will result. Level 1 printers and emulators cannot use 'setpagedevice' and will also report an error. Insert the language level conversion on page 63 or remove the line and substitute 'a4', 'letter', or 'legal'.

% *PostScript Markup*

% 43 %



Most laser printers deliver pages face down, unless there is a rear door that allows face up stacking. For the first printing you may have to turn each one over as it emerges and stack it in the correct order before printing the reverse. Wedge a piece of half-round plastic guttering in the exit slot to curl the first side printed pages backwards out of the machine.

### Custom Paper Sizes

```
<<
/DeferredMediaSelection true          % bypass printer defaults
/MediaPosition 1                     % 1 = multipurpose tray; 0 = tray 1
/Policies <</PageSize 5 >>         % do not adjust print to suit paper size
/PageSize [ 612 1008 ]               % *legal*: insert your choice
/ImagingBBox null                    % maximum printable area
/shift { 0 170 translate } def       % optional adjustment: insert into page format
>> setpagedevice                    % instructions may be grouped together
```

PostScript printers will sulk if the paper being used does not match the on-board default size or tray selection. Make a little file headed by ‘%!PS’ and ‘true 0 startjob’; paste in the above instructions, insert the required paper size and download to the printer. If ‘shift’ is typed after `_Z` in the page format, A4 sheets will print head-up on a tabloid printer when legal is selected.

Do note that the manufacturer’s tray number varies from model to model and you may need to refer to the handbook for the Media Position number.

For multiple copies in collated order, e.g. 1,2,3 : 1,2,3 use:

```
<</NumCopies 1 /Collate true>> setpagedevice          % PostScript 3
The number of pages capable of being collated depends on the amount of
memory available in the printer.
```

### Duplexing

```
<< /Duplex true /Tumble true >> setpagedevice
% unchecked to use: tumble true = short edge
2upA4 folios may be duplex printed by file merging front and back
PDF pages in Adobe Exchange. You may re-impose 1upA5 pages in
2upPP format as printer’s pairs; using the pdf2folio conversion; or
spooling with a document manager. See section 10.
```

## ERRORS

PostScript definitions begin with a foreslash, have a name, are enclosed in a pair of { braces }, and end with ‘def’. An exception is a data statement such as ‘year 2003 def’. All text before a paragraph return must be enclosed between delimiting (parentheses).

Looking for errors is like Hunting the Snark, so use a spelling checker customized to include the Tinydict codes and so identify misprints. Most errors result from our own stumbled typing. The usual infelicities are a misspelling; a missing code; a omitted space such as ‘9rom’, or an unmatched pair of { } braces or ( ) text delimiters.

<b>add / sub</b>	an ‘a’ or ‘b’ movement without a value.
<b>defaults</b>	unpaired <code>_Z</code> : unused data: or more than one ‘close’ in the file.
<b>dictfull</b>	increase Tinydict memory for level 1 interpreter.
<b>dup</b>	linewrapping codes J, CS, S or P have no text or a Distiller filepath is inserted after ‘place’ instead of before.
<b>end</b>	Tinydict or other dictionary missing
<b>exch</b>	missing data: e.g. ? rom: ? TM: # ? # place.
<b>filetype</b>	undefined or incomplete instruction
<b>findfont</b>	font not downloaded, or not in correct font folder.
<b>front / back</b>	Tinydict not in printer or Distiller memory. Paste it in the file prolog OR uncheck ‘startjob’ and download beforehand.
<b>do</b>	Distiller response to undefined instruction or an unpaired )
<b>gt</b>	an unnecessary ‘nocols’.
<b>invalid access</b>	level 1 ‘exitserver’ is in a printing file.
<b>jump</b>	‘text’ instruction without a previous ‘place’.
<b>no currentpoint</b>	the textbox co-ordinates are lost.
<b>no stringval</b>	a missing ‘} def’ at the end of a definition.
<b>not found</b>	font name not spelt accurately
<b>pages</b>	remove space: use ‘2pages’, not ‘2 pages’ or ‘4 pages’.
<b>PH / PW</b>	as front / back
<b>setpagedevice</b>	non-standard paper sizes sent to printer or, if level 1, prefix a % and use ‘a4’ or ‘letter’.
<b>show</b>	coloured text without CS or a code with no text.
<b>search</b>	M, J, CS, S or P have no text .
<b>status</b>	Distiller error: unused data on stack: e.g. 9 without ‘rom’.
<b>stringwidth</b>	data above text: misplaced colour e.g. ‘blue centre CS’.
<b>tiledown / up</b>	insert a front and back toggle for folio printing.
<b>typecheck</b>	a misspelling, undefined instruction, or unknown code
<b>undefined</b>	missing ( or ) text delimiter or unknown code.

If a page textbox appears in the wrong place. the ‘CS’ colour code has been used across a page jump.





## PDF RESOURCES

Portable Document Format files are normally made by saving a desktop printing or graphics document as a PostScript file from the Print dialog. The file is then converted into PDF by Distiller and opened in the Exchange portion of the Acrobat trilogy. Bookmarks, links, and action buttons are added by selecting the appropriate menu item; clicking a suitable point in the on-screen page; typing the text required, and moving on to the next one. This lengthy process inserts what are known as pdfmark instructions in the file to provide the necessary links and bookmarks when the distilled PDF document is opened.

Bookmarks make the search for relevant material faster and easier, whilst links allow jumps to other PDF pages, files, or Web pages to be made instantaneously. The great disadvantage is that any re-distilling of the original PostScript file means that the PDF file will be overwritten and the whole process done again.

Tinydict mark-up files are ideally suited for creating PDF pages because the bookmarks and links remain permanently in position; do not have to be re-inserted for a re-distilled file; and remain editable at any time.

As a Tinydict typeset file is designed to be either printed by laser or distilled by Adobe Acrobat, the following instruction is required in the prolog of a self-printing file, so that any pdfmarks in the file do not give the PostScript interpreter in the laser printer digital indigestion. `/pdfmark where {pop} {userdict /pdfmark /cleartomark load put}` Older level one PostScript printers need some further medication by using: `/languagelevel where {pop languagelevel} {1} ifelse 2 lt {userdict {<<} cvn (l) cvn load put userdict {>>} cvn (l) cvn load put} if` A Tinydict file for conversion into a PDF document should also include the following information pasted into the file header. Amend the text as required.

```
[ /Title (Tiny Resources)
/Author (David Byram-Wigfield)
/Subject (PostScript Typesetting)
/Keywords (Typesetting Mark-up)
/Creator (Tinydict 5 )
/ModDate (D:14.3.2003)
/DOCINFO pdfmark
```

To ensure the PDF opens with bookmarks or thumbnails use:

```
[ /PageMode /UseOutlines /DOCVIEW pdfmark
[ /PageMode /UseThumbs /View [ /XYZ -5 797 1.2 ] /DOCVIEW pdfmark
```

## BOOKMARKS

PDF bookmarks may be added anywhere in a Tinydict typeset file. A positive or negative Count indicates the number of related nested bookmarks and whether they are open or closed. The first article in this list has two closed bookmarks and the second one would be followed by five related bookmarks, all of them open.

```
[ /Count -2 /Title (First Article) /OUT pdfmark
[ /Title (subject 1) /OUT pdfmark
[ /Title (subject 2) /OUT pdfmark
[ /Count 5 (Second Article) /OUT pdfmark
[ /Title (subject 1) /OUT pdfmark % repeated five times
```

## NOTES

A PDF note is positioned by embedding suitable rectangular co-ordinates in lower-left upper-right order. There is an open true or false instruction; a title; a string of text; a titlebar and optional background colour in rgb values; and a closing announcement label.

Fortunately, the only rectangular dimension requiring input in a Tinydict file is the depth needed for the note, which is the required amount of drop subtracted from the current text position `tm` on the page. The width may be set to a proportion of the textwidth `rm`, or any other chosen value. As an opened note will obscure any underlying text, it may be embedded as a closed note by the instruction 'false'.

For example, a full textwidth closed note with a 100 point drop and a red titlebar and would be set as:

```
[ /Rect [ 0 tm 100 sub rm tm ] /Open false /Title (A Full-width Note)
/Contents (A sample full-width note ) /Color [ 1 0 0 ] /ANN pdfmark
```

A centered half-width Rect would read:

```
[ /Rect [ rm 0.25 mul tm 100 sub rm 0.75 mul tm ]
```

Sometimes a narrow closed note is needed outside the text area in the left margin. It is made by giving the first Rect x co-ordinate a minus value and inserting a narrow width, such as 30 points, in place of `rm`. `[ /Rect [-30 tm 30 sub -10 tm ]`





## LINKS AND ACTIONS

A variety of pdf links is available for jumping pages; finding other files; opening URLs, etc.; and the following are only a few examples. Whilst there are methods of moving between local pages, such as 'Prev' and 'Next', for long-distance jumping insert a destination name on the page to be jumped to, like this:

```
[ /Dest /Pdfmarks /DEST
```

The destination here is 'Pdfmarks', which is better than calling it /Page\_14, as any later editing may alter the numbering sequence. We then define a matching link to jump to our destination PDF. The textbox current line tm will place the link box level with related text, whilst minus 35 moves it into the left hand margin of the page. Do not put a zero value for the right hand horizontal position or you may get an error message.

```
[ /Dest /Pdfmarks /Rect [ -35 tm 20 sub -15 tm ] /View [ /XYZ 400 400 1.2 ]  
/Border [ 0 0 2 ] /Color [ 1 0 0 ] /Subtype /Link /ANN
```

The /View mark may be omitted, its XYZ values determine the area of the page to be shown after the jump (0 being the default), whilst the Z value is the amount of zoom magnification; 1 being 100%. The Border gives a thick, thin, or dashed outline, and Colour specifies the rgb values of the outline of the link box.

A link that launches another pdf file is a quick way of opening a related document.

```
[ /Rect [ -35 tm 20 sub -15 tm ] /Border [ 0 0 2 ] /Color [ 1 0 1 ]  
/Action /Launch /File (myfile.pdf) /Subtype /Link /ANN
```

A link that creates an action for an Acrobat plug-in, such as opening a web site, has to include pairs of << >> lesser and greater than markers.

```
/Rect [ -35 tm 20 sub -15 tm ]  
/Action << /Subtype /URI /URI (http://www.ifconnection.de/~tm/) >>  
/Border [ 0 0 2 ] /Color [ 0 1 1 ] /Subtype /Link /ANN pdfmark
```

See the primer by Thomas Merz: *Web Publishing with Acrobat PDF*  
ISBN 3 540 63762 1: published by Springer

<http://www.pdfzone.com/>

## PDF NAVIGATION

The TinyLinx creates page links like those in these pages.

```
%%BeginResource:tinylinx  
/pointer { np 0 0 mt 8 14 li 16 0 li cp } bind def  
/gopos { rm 2 div 8 sub } def  
  
/gofind { gs gopos -68 tr  
pointer 0 0 1 setrgbcolor fill  
[/Rect [ 0 -5 35 32 ] /Border [ 1 0 0 ] /Page 1  
/View [ /XYZ null null null ] /LNK pdfmark gr } def  
  
/goback { gs gopos 40 sub -62 tr  
30 neg rotate pointer 1 0 0 setrgbcolor fill  
[/Rect [ 0 -5 35 32 ] /Border [ 1 0 0 ] /Page /Prev  
/View [ /XYZ null null null ] /LNK pdfmark gr } def  
  
/gonext { gs gopos 40 add -70 tr  
30 rotate pointer 0 1 0 setrgbcolor fill  
[/Rect [ 0 -5 35 32 ] /Border [ 1 0 0 ] /Page /Next  
/View [ /XYZ null null null ] /LNK pdfmark gr } def  
% insert tinylinx in the format textbox  
/tinylinx { goback gofind gonext } bind def  
/textbox { margins tinylinx bodytext } def  
%EndResource
```

The 'gofind' refers to any page in the file. The example of Page 1 quoted will return the user to the first page of the PDF document which may not be the first numbered page.



## FONT RE-ENCODING

A PostScript font encoding may contain up to 255 character outlines, each of which has its own descriptive name, such as ‘acute’, and an identifying PostScript octal number. Many of the characters do not appear in a standard operating system encoding, or are not mapped onto the correct keys. A character will only appear in a line of text if it has been encoded onto a particular key, or by using its octal number. <sup>1</sup> For instance, an em–dash may be printed using the octal \320 to produce this–.

For the benefit of beginners, ‘encoding’ means allocating the name of a character outline (sometimes known as a ‘glyph’) to an ascii number, usually between 32 and 255, as 0 to 31 are needed for on–screen controls, such as carriage return, page up or down, etc. The ascii numbers are in their turn ‘mapped’ onto a particular key, which varies between computer operating systems and foreign languages.

For world–wide compatibility the ascii numbers 32 to 127 are reserved for the roman alphabet, numerical figures, punctuation, etc. The sequence 128 to 192 is ignored in the Macintosh Standard Encoding, apart from a few ‘dead key’ accents, although many PC characters live there. The remaining 193 to 255 usually hold an ISO–1 or 2 encoding for foreign languages.

I have made a Custom Encoding resource for the Tinydict so that the ISO–Latin1 accented characters are available. For English speakers, the re–encoding puts the single and double curly ‘smart’ quotation marks onto the option (alt) and option–shifted { } and [ ] keys used in normal word–processing. To encourage the proper use of the en–dash instead of the hyphen between figures like 1998–9, I have keyed the em– and en–dashes in the place of the Eth and Ntilde which normally live on the ISO option and option+shift hyphen: my apologies to any ?Hungarians.

*If you Distil a re–encoded font using Acrobat 3.01 you may find that some of the higher characters are incorrect when they appear on the screen, although the page will print correctly on a PostScript printer. This does not occur with later versions of that software.*

---

1. Octal numbering counts in base 8, starting with \000 as 0, so \007 represents 7, \010 is 8 and \017 is 15. The ascii text characters start with the space at octal \040, which in normal counting is decimal 32.

%!PS

%%BeginResource: isocode1.ps

%%Creator: David Byram–Wigfield

%%For: Cappella Archive

%%Revision Date: 15 February 01

%%EndComments

% None of the % commented lines must linebreak

% Insert this Resource into Section 36 of the Tinydict

% Re–encodes Adobe Standard PostScript as Mac ISO–Latin 1

% for smart quotes and western European languages

% PCs may be able to use the octal numbers shown

% Create an empty array

/CustomEncoding 256 array def

CustomEncoding 0

% insert ISO into empty Custom encoding

ISOLatin1Encoding 0 255 getinterval putinterval

% duplicate the array for the next one

CustomEncoding

% Re–allocate a character for normal WP keying

dup 210 /quotedblleft put % Mac option [ (\147 PC)

dup 211 /quotedblright put % Mac option/shifted [ (\148 PC)

dup 212 /quoteleft put % Mac option ] (\145 PC)

dup 213 /quoteright put % Mac option/shift ] (\146 PC)

% allocate the dashes and decimal point

dup 208 /endash put % Mac option – (\150 PC)

dup 209 /emdash put % Mac option/shifted – (\151 PC)

179 /periodcentered put % Mac option . (fullstop or point)

% no duplication for the last one

% copy everything except the FID Number

/customfont {

findfont dup length dict begin {1 index /FID ne { def }

{ pop pop } ifelse } forall

/Encoding CustomEncoding def currentdict end definefont pop

} bind def

/F { exch selectfont } def

%/F { findfont exch scalefont setfont } def

% uncheck % for level 1



```
% use correct PostScript fontnames: download non-printer fonts first
% ss = sans serif: change size: e.g. 24 rom, 9.5 it, 12 ss, etc
/rom {/_Times-Roman dup /Times-Roman customfont F } bind def
/it {/_Times-Italic dup /Times-Italic customfont F } bind def
/bol {/_Times-Bold dup /Times-Bold customfont F } bind def
/bolit {/_Times-BoldItalic dup /Times-BoldItalic customfont F } bind def
/ss {/_Helvetica dup /Helvetica customfont F } bind def
/sb {/_Helvetica-Bold dup /Helvetica-Bold customfont F } bind def
/si {/_Helvetica-Oblique dup /Helvetica-Oblique customfont F } bind def
%%EndResource
```

Some octals are mapped onto Macintosh option and option + shifted keys. PC users may insert the octal number into a text string by using the alt key and numeric keypad. For example, Fran\347oise prints Françoise. Common accents are: \350 è \351 é \352 ê \353 ë

The empty sequence \200 to \207 and \210 to \217 may be used to add the Adobe PDF encoding or for specialist font characters or logos. Macintosh option+shift A and C access octals \201 and \202, whilst option a and c similarly enable \214 and \215.

### Accents

Instead of re-encoding, accents may also be inserted in level 2 or above by using 'glyphshow'. However, you need to know the encoding name of the character, and it may not be always present in the font.

```
(f) show /ecircumflex glyphshow (te) show = fête
(PostScript) show /trademark glyphshow = PostScript™
```

### Changing Styles

The Tinydict can mix styles on the fly; use either 'j' or 'S'

```
9 rom (This is Times-Roman, ) j 9 it (and this is italic. ) j 9 rom (You may include ) j
9 bol (bold ) j 9 rom (and then ) j 9 bolit (some bold italic. ) j L
```

This is roman, *and this is italic*. You may include **bold** and then *some bold italic* and finally back to roman again.

### Re-encoded Times-Roman - Octal Numbers

#### Control keys

000	001	002	003	004	005	006	007
nul	home	ctrl-b	enter	end	help	ctrl-f	ctrl-g
010	delete	tab	ctrl-j	p-up	p-dwn	return	ctrl-n
020	ctrl-p	ctrl-q	ctrl-r	ctrl-s	ctrl-t	ctrl-u	ctrl-v
030	ctrl-x	ctrl-y	ctrl-z	esc	lh<	rh>	up
							down

#### Ascii Standard Encoding - 7-bit alpha-numeric

040	space	!	"	#	\$	%	&	'
050	(	)	*	+	,	-	.	/
060	0	1	2	3	4	5	6	7
070	8	9	:	;	<	=	8	?
100	@	A	B	C	D	E	F	G
110	H	I	J	K	L	M	N	O
120	P	Q	R	S	T	U	V	W
130	X	Y	Z	[	\	]	^	_
140	`	a	b	c	d	e	f	g
150	h	i	j	k	l	m	n	o
160	p	q	r	s	t	u	v	w
170	x	y	z	{		}	~	

#### Re-encoded with 8-bit Latin-ISO 1

200								
210								
220	ı	˘	˙	˚	˛	˜	˝	˞
230	˜	˚	˛	˝	˞	˟	ˠ	ˡ
240	ˡ	ˢ	ˣ	ˤ	˥	˦	˧	˨
250	˨	˩	˪	˫	ˬ	˭	ˮ	˯
260	˯	˰	˱	˲	˳	˴	˵	˶
270	˶	˷	˸	˹	˺	˻	˼	˽
300	À	Á	Â	Ã	Ä	Å	Æ	Ç
310	È	É	Ê	Ë	Ì	Í	Ï	Ñ
320	—	—	“	”	‘	’	Ö	×
330	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
340	à	á	â	ã	ä	å	æ	ç
350	è	é	ê	ë	ì	í	î	ï
360	ð	ñ	ò	ó	ô	õ	ö	÷
370	ø	ù	ú	û	ü	ý	þ	

Octals 321 to 325 are encoded here with en and em dashes, double speech marks and single quotation marks.



## THE TINY TABLES

```
%%BeginResource:tinytables
% Insert Resource in Section 37 of the Tinydict
/FW { /tabs exch def } def 75 FW % default field width
/NUM { /num exch def } def 100 NUM % default field number up to 1000
% insert incremental number into a field: e.g. fldnum lf or lb
/fldnum { num num 1 add NUM 4 string cvs } def
/ef { lm tm moveto tabs lm add LM } bind def % empty tab advance
/eb { tb ef } bind def % empty tabbed box
% tab with left handed text : e.g. (text) lf or lb
/lf { ef 2 0 rmoveto s } bind def % tab with left handed text : e.g. (text) lf
/lb { tb lf } bind def % boxed tab with left handed text : e.g. (text) lb
/cf { ef dsp 2 div tabs 2 div exch sub 0 rmoveto s } bind def
/cb { tb cf } bind def % tab with centered text: e.g. (text) cf or cb
% tab with right handed text: (text) rf or rb
/rf { ef dsp tabs exch sub 0 rmoveto 2 neg 0 rmoveto s } bind def
/rb { tb rf } bind def
% tab with decimal text: e.g. (3) df or db (.1416) s
/df { ef dsp tabs 2 div exch sub 0 rmoveto s } bind def
/db { tb df } bind def
% newline after last tab: resets textbox left margin
/nl { 0 LM L } bind def
% tabbed box
/tb { gs newpath lm tm moveto 0 lg 4 div neg rmoveto 0 lg rl tabs
  0 rl 0 lg neg rl closepath 0.1 set grestore } bind def

% calculation procedures: numbers default to two decimal places
% convert real number to text string
/convert { 100 mul round 100 div 10 string cvs } bind def
% mark items to be calculated: e.g. (150) * lh: (29.50) vat * rh
/* { dup cvr 100 mul round 100 div exch } bind def
% @ multiply previously * marked items and store the result
/@ { mul * convert } def
% total add * marked items
/total { count 1 sub { add } repeat convert } bind def
% vat calculation: cvr = convert text to real number
% to add vat to a price type vat add : alter value for variations
/vat { cvr dup 0.175 mul convert } bind def
%%EndResource
```

## TinyTable Codes

# NUM	default first incrementing number
# FW	variable field width: alter LG for different depth
ef or eb	empty field: boxed empty field.
lf or lb	left handed text field: boxed left handed text field
cf or cb	centered text field: boxed centered text field
df or db	decimal number field: boxed decimal number field
rf or rb	right handed text field: boxed right handed text field
nl	newline: use at end of every row
*	mark field entry for calculation
@	multiply the last two marked fields
vat	calculate nearest marked field
total	add together marked fields
fldnum	insert incrementing number
clear	remove any lost numbers: type after completing a table or a sequence of related tables.

The depth of each field is the same as the LG text linespacing and the field width is defined by a FW variable, set by default at 75 points. Both may be changed on the fly to any other value. The field justification codes are either plain or boxed using, for example, rf or rb. It is important to complete each row by typing nl for new line or the left hand textbox margin will not re-set.

Fields to be calculated are given a \* marker which may also be placed after any calculation but before the field justification code. The vat instruction will calculate the amount of the nearest previously marked field and vat itself may be marked for a total field to be added.

Each multiplication automatically places a marker to provide a total in a following field. The fldnum instruction provides a chosen default number and increments it by 1 for numbering a sequence of invoices. The value is incremented until the file is closed and may be reset at any time. Remember to type 'clear' to remove any duplicated numbers after making tables. A field width only needs re-setting after any alteration.



## TinyTable Field Codes

<b>ef = empty field</b>	<b>lf = left field</b>	<b>df = decimal field</b>	<b>cf = centered field</b>
<b>rf = right field</b>	<b>nl = nextline</b>	<b>eb = empty box</b>	<b>lb = left box</b>
<b>db = decimal box</b>	<b>cb = centre box</b>	<b>rb = right box</b>	<b>fldnum = number</b>
<b>vat = x 0.175</b>	<b>* = marker</b>	<b>@ = multiply *</b>	<b>total = add *</b>

### Plain Fields

<b>left-handed</b>	<b>decimals</b>	<b>centered</b>	<b>right-handed</b>
<b>some</b>	<b>price</b>	<b>vat</b>	<b>total</b>
<b>examples</b>	<b>23.27</b>	<b>4.07</b>	<b>27.34</b>
<b>43.20</b>	<b>93.12</b>	<b>16.3</b>	<b>152.62 row total</b>

% first invoice number: default field width: 9 pt sans serif: 10pt linespacing

**230 NUM 75 FW 9 ss 10 LG**

**(left-handed) lb (decimals) cb (centered) cb (right-handed) rb nl**

**(invoice no.) lb (price) df (quantity) cb (value) rb**

**fldnum lb (0.87) \* db (2300) \* cb @ rb nl**

**ef (1.35) \* db (250) \* cb @ rb nl**

**ef ef (column total ) cb total rb nl**

### Boxed Fields

<b>left-handed</b>	<b>decimals</b>	<b>centered</b>	<b>right-handed</b>
<b>invoice no.</b>	<b>price</b>	<b>quantity</b>	<b>value</b>
<b>230</b>	<b>0.87</b>	<b>2300</b>	<b>2001.0</b>
	<b>1.35</b>	<b>250</b>	<b>337.5</b>
		<b>column total</b>	<b>2338.5</b>

Altering the LG value determines the depth of the field and its box. Adjusting FW will vary the field width and ef empty fields will tab where required on the page. For example:

**14 LG 40 FW 10 ss ef (test) cb 20 LG 60 FW (extra large) cb nl**

test	extra large
------	-------------

## MARK-UP FOR CALCULATION FIELDS

Notice that the \* marker is placed after any calculation but before the field code and numbers usually look better placed using the rf or rb codes. The multiplication marker @ duplicates the result and a following total field must be provided in which to put it. Total and vat fields may themselves be marked \* to store the value for a final total as shown below. The field width only needs re-setting after any alteration and the vat value may be altered for local variations.

**14 ss (LIMBO DIGITS LTD.) c % centered**

**8 sb 14 LG A** % 8 pt sans serif: 14 pt linespacing: half line advance

**(Invoice No.) cb (Date) cb (Order No.) cb (Date Ordered) cb nl**

**(2905) cb (5 Sept 98) cb (XY5093) cb (2 Jan 98) cb nl**

**ef (item) cb 20 FW (qty) cb 60 FW (price) cb (value) cb nl**

**70 FW ef (widgets) lb 20 FW (3) \* cb 60 FW (29.45) \* db @ rb nl**

**70 FW ef (corkscrews) lb 20 FW (70) \* cb 60 FW (4.92) \* db @ rb nl**

**70 FW ef (stoppers) lb 20 FW (90) \* cb 60 FW (0.79) \* db @ rb nl**

**50 FW ef 40 FW ef 60 FW (sub-total) cb total \* rb nl**

**50 FW ef 40 FW ef 60 FW (vat) cb vat \* rb nl**

**50 FW ef 40 FW ef 60 FW (total) cb total rb nl**

**clear**

Always type 'clear' at the end of the table codings, unless values are being transferred to a following one. In that event, type 'clear' after the last table just in case there are any uncalculated numbers left on the PostScript stack.

### LIMBO DIGITS LTD.

<b>Invoice No.</b>	<b>Date</b>	<b>Order No.</b>	<b>Date Ordered</b>
<b>2905</b>	<b>5 Sept 98</b>	<b>XY5093</b>	<b>2 Jan 98</b>

<b>item</b>	<b>qty</b>	<b>price</b>	<b>value</b>
<b>widgets</b>	<b>3</b>	<b>29.45</b>	<b>88.35</b>
<b>corkscrews</b>	<b>70</b>	<b>4.92</b>	<b>344.4</b>
<b>stoppers</b>	<b>90</b>	<b>0.79</b>	<b>71.1</b>
	<b>sub-total</b>		<b>503.85</b>
	<b>vat</b>		<b>88.17</b>
	<b>total</b>		<b>592.02</b>



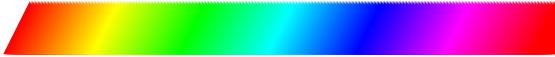
## USING COLOUR

There are four PostScript colour modes. `Setrgbcolor` relates to the red, green and blue primary colours required for the monitor display of PDF and Web pages. The `setcmykcolor` reflective colours create the separations necessary for offset colour printing, whilst `sethsbcolor` also sets red, green, or blue colours, but as a hue; varying the saturation and brightness like the controls on a television set.

The grayscale mode applies a decimal percentage shades of gray from black to white using the `setgray` operator. Each multi-colour mode may be converted into either of the other two.

The rainbow shows a comparison between `rgb`, `hsb`, `cmk`, and the Adobe `setgray` complementary values. The nearer all three colours are to zero in `rgb`, `hsb`, and the single colour of `setgray`, the nearer they are to black; whereas the four `cmk` colours become lighter. The gray values shown are those adopted by Adobe to represent the adjoining colour when a colour file is printed on a monochrome printer. Native English-speakers need to beware the American spellings.

	Red	Yellow	Green	Cyan	Blue	Magenta	Red
<code>setrgbcolor</code>	1 0 0	1 1 0	0 1 0	0 1 1	0 0 1	1 0 1	1 0 0
<code>setcmykcolor</code>	0 1 1 0	0 0 1 0	1 0 1 0	1 0 0 0	1 1 0 0	0 1 0 0	0 1 1 0
<code>sethsbcolor</code>	0 1 1	0.16 1 1	0.33 1 1	0.5 1 1	0.66 1 1	0.83 1 1	1 1 1
<code>setgray</code>	0.3	0.89	0.59	0.7	0.11	0.41	0.3



The colours in the Tinydict are `cmk` values, although some basic `rgb` abbreviations are provided for those who need them. The naming is a matter of personal taste, but colour changes should be easily identified in a marked-up file.

**(Use CS ) S (like this ) blue CS (to colour words in a line of text. ) P**

Don't forget the space before the final bracket and add an L line advance to move to a new line. To colour an entire paragraph prefix with, say, 'blue `cmk`' and restore the black with '0 `setgray`' afterwards.

## SECTION NUMBERING

```
%%BeginResource:secnumb.ps
/SEC { /sec exch def } def 1 SEC
/NUM { /num exch def } def 1 NUM
/nextsec { sec 1 add SEC 1 NUM } def
/putnum { num num 1 add NUM 4 string cvs show } def
/SN { gsave 8 sb -25 T sec 4 string cvs show
(.) show putnum grestore } bind def
/revert {1 SEC 1 NUM } def
%%EndResource
```

SN (Some users may like a sectioning mark-up Resource for the Tinydict. ) P  
SN (These section numbers are offset 25 points to the left by -25 T. ) P  
SN (Modify the Tinydict numbering procs for odd/even page footer or right hand placing. ) P  
nextsec  
SN (Use 0 T for a flush left start and increase LM to indent the text. ) P  
SN (New section. ) P  
SN (And so on. ) P  
revert  
SN (Start again ) P  
24 NUM  
SN (Be ambitious) P

- 1.1 Some users may like a sectioning mark-up Resource for the Tinydict.
- 1.2 These section numbers are offset 25 points to the left by -25 T.
- 1.3 Modify the Tinydict numbering procs for footer or header placing.
- 1.4 Use 0 T for a flush left start and increase LM to indent the text.

```
/SN { gsave 8 sb 0 T sec 3 string cvs show
(.) show putnum grestore } bind def
```

Modify LM but don't forget a slight movement: e.g. 30 LM MT  
30 LM MT

- 2.1 New section.
- 2.2 And so on.
- 1.1 Start again
- 1.24 Be ambitious





## SETTING EQUATIONS

```

%%BeginResource:equation.ps
% superior and inferior figures and linespacing are in Tinydict
/half { lg 2 div } bind def % half linespacing
/quarter { lg 4 div } bind def % quarter linespacing
/k { 0 rmoveto } def % horizontal kerning: e.g. 2 k or -5 k
/v { 0 exch rmoveto } def % vertical kerning: e.g. 5 v or -10 v
/up { half v } def % move up half the linespacing
/dn { half neg v } def % move down half the linespacing
/N { up msl show dn } bind def % raise: measure and place numerator: lower
/D { half neg v msl show up } bind def % do the opposite for the denominator
/ML { /ml exch def } def 0 ML % store measured length
/msl { dup stringwidth pop ML } def % measure stringlength of text
/bar { gsave 3 v neg 0 rlineto lg 0.05 mul setlinewidth stroke grestore } bind
def
/BK { ml neg k } def % move back bar length
/MB { ml bar BK } def % place bar automatically *after* N or D
/PB { ml mul bar } def % bar proportional to *last* N or D: e.g. 1.5 PB
/makebrace { currentpoint half sub moveto 0 0 -6 lg 2 sub 0 lg half add
rcurveto lg 0.05 mul setlinewidth stroke } bind def
/DLB { gsave makebrace grestore } def
/DRB { gsave -1 1 scale makebrace grestore } def
/sym {/Symbol F } def % define Symbol font
/sqroot {/Symbol F ( ) s } def % square root octal number
%EndResource

```

This is a rudimentary resource for setting basic equations. At present, it relies on the Symbol font for mathematical symbols, which is fine for Macintosh owners, but really needs its own maths font to make the 8-bit characters available to other systems. Three criteria are important. First, the mark-up file itself should be capable of interpretation to the average eye. Secondly, vertical measurements should be proportional to the chosen linespacing, so that the equation remains evenly spaced vertically, irrespective of fontsize. Thirdly, horizontal measurements should be proportional to the fontsize because fixed distances in points would cause linear distortion if the fontsize were changed.

An equation usually has three fractional units; the numerator; the division line; and a denominator. The numerator and denominator may themselves be similarly divided into the same three components and any character may also have an attached superior or inferior figure. In addition, mathematical symbols are needed for the square root, the Greek alphabet, braces, etc.

Let's start with an easy one;  $2x / y = 24$ . The longer of the numerator or denominator is set first with an N or D code, in this case the numerator 2x. MB draws the correct length of bar and moves the denominator to the left, which is centred by a space on either side and set by D. The centre line characters are set by the in-line s code.

$$(2x) N MB ( y ) D ( = 24 ) s \text{ becomes } \frac{2x}{y} = 24$$

If the longer unit is broken by superior or inferior figures or by in-line fractions, then the despised word processing method of formatting with spaces is used either side of the shorter unit and that is set first. A superior figure needs an extra lift with up and dn codes, or it will be set relative to the bar, not the associated upper or lower character.

$$( y ) D MB (2x) N up (2) sup dn ( = 24 ) s \text{ becomes } \frac{2x^2}{y} = 24$$

The expression:  $3x \text{ sqd} / 4y + 5x = 24$  is set with the longer denominator first, then the bar, and the numerator.

$$(4y + 5x) D MB ( 3x) N up (2) sup dn ( = 24 ) s \frac{3x^2}{4y + 5x} = 24$$

The current body fontsize/style (in this instance 10 point roman) must be reset after using the Symbol font. The proportional bar PB, which sets the the overall numerator square root, is relative to the length of the last D or N and may need fine-tuning after proofing. A T tabbing value moves the equation across the page.

200 T (4y + 5x) D MB up ( ) s 12 sqroot dn 9.5 rom ( 3x) N up (2) sup dn up up 1.2 PB dn dn ( = 24 ) s

$$\frac{\sqrt{3x^2}}{4y + 5x} = 24$$



Sometimes left and right handed curved double braces are needed to span both the numerator and denominator. As they aren't in the Symbol font, we have to make our own DLB and DRB. They look jagged on-screen in PDF but print properly. Any following text may need spacing into the correct starting position.

90 T (4.25) D MB ( 1 ) N ( x ) s DLB 2 k ( 273 ) N MB ( 183 ) D DRB up ( 2 ) sup dn ( x ) s (5) N MB (6) D ( = )s (5.5) D MB ( 1 ) N ( or nearly. ) P

$$\frac{1}{4.25}x \left( \frac{273}{183} \right)^2 x \frac{5}{6} = \frac{1}{5.5} \text{ or nearly.}$$

Setting text is quite easy:

90 T (mitre length) N MB ( line width ) D ( = ) s ( 1 ) N MB dn dn (sin ) s DLB 10 sym ( j ) N 9.5 rom DRB MB ( 2 ) D up up

$$\frac{\text{mitre length}}{\text{line width}} = \frac{1}{\sin\left(\frac{\phi}{2}\right)}$$

This equation resource really needs a Maths font for an italicised alphabet and the correct minus and multiplication signs and period decimals.

## AUTOMATIC FOOTNOTES

% insert these procs in the Tinydict section 37

%%BeginResource:footnote.ps

% change fontsize/style and linespacing to suit:

% FN = start footnotes: raise bottom text margin:

% the \_Z - ZZ pair preserve existing format and current point

% set 8 pt roman on 10 pt spacing: place rule

**/FN { bm add BM \_Z 8 rom 10 LG Im bm TM tm moveto HR } def**

**/EN {ZZ 9.5 rom 13 LG } def**

% EN = end footnotes: restore 10 pt roman on 13 pt spacing

% amend fontsize and spacing to suit chosen values in textbox

%%EndResource

Insert the footnote text before the reference line or paragraph.

Guess the number of lines of text and correct the FN value after proofing.

**54 FN**

% raise BM: footnote linespacing x no. of lines\*

**(1. Text of footnote here ) p**

% non-page jumping code

**(2. Next same page footnote ) p**

% or use nj for full justification

**EN**

% restore bodytext

Insert the reference number in the text after the footnote: e.g.

**( . . target word ) j or S (1) sup ( add a space before the remaining text.) P**

The distance you raise BM depends on the number of lines of footnote text and linespacing. <sup>1</sup> You may add a few points to position the rule <sup>2</sup> midway between the body text and the footnotes. After a few examples, you should be able to calculate a list of suitable line values. Insert all closely adjoining footnotes as one unit <sup>3</sup>

The body text will autoflow from page to page leaving sufficient space for the footnotes, which are automatically inserted in the right place. If the page formatting is changed, the footnotes will still travel with the text. However, a footnote reference in the last two lines on a page is likely to have the corresponding footnote printed on the next.

---

1. One method of calculating footnote depth in advance is to paste all the footnotes into one file and then apply a 1upA5 format with footnote fontsize and linespacing and print out.

2. Delete HR horizontal rule if not needed

3. If not, they will appear in reverse order!



## GREP MARK-UP PATTERNS

These Grep patterns convert Ascii text into a PostScript marked-up script. It is a very fast method of typesetting copy as book pages for pagination proofs. BBEdit on a slothful classic Macintosh can mark-up a 1 megabyte ascii file in 20 seconds. (191,000 words: 452 paragraphs: 1,049,558 characters)

1. Use a text editor with a line unwrapping facility and a Grep pattern search such as NotePad Pro, BBEdit, or Alpha.
2. Open *a copy* of the Ascii text file to be typeset
3. Open the menu: Text or Special or Search->Find:
  - a) Replace left hand curly quotes with grave (shift+tilde)
  - b) Replace all right hand curly quotes with straight apostrophe.  
**Ignore a and b if your chosen font has double quotes encoded**
  - c) Prefix all ( and ) parentheses in the text with a backslash (Make sure there is a matching number).
4. Under Edit/Window Options, etc: unwrap the paragraphs into single lines. Long paragraphs will disappear off the screen to the right .
5. Select a section or check Cursor to Top (as applicable)
6. Open the menu: Text or Special or Search->Find: click Grep button. Create the following patterns:

```
Find ^(.+)$           % note the caret (shift-6): no spaces please
Replace (\1 ) J       % justify: note the space before the final parenthesis
For ragged right setting use:
Replace (\1 ) P       % flush left: note the space before the final parenthesis
```
7. Then Replace All
8. Insert 'newpage' for chapter or section page breaks and 'close' at end. See section 21.
9. Remove empty ( ) lines. Centre chapter header titles with 'c'.
10. Insert font/size changes where required. **Restore bodytext for following text**
11. Distil into PDF or add a 16up TinyImp header and print a proof. (See section 21 for setting chapter breaks)

*The Adobe Standard PostScript Encoding needs re-encoding to print double smart quotes and sterling pound signs. See the Tiny Resource for 8 bit ISO-1 Encoding.*

## PostScript Reversal

An interesting use for Grep is to extract the text from an existing PostScript file, which under normal editing circumstances is almost impossible. Open a copy of the PostScript file in a text editor with a Grep facility such as BBEdit (Mac) or NoteTab (PC) Insert:

```
Find (.[^\[]*)(.[^\]]*)
Replace \2 \r
```

There must be a space between the \2 and the \r return.

Here is some script:

```
(All rights r)Tj [9.7 ]TJ
( eser)Tj [-18 ]TJ
(v)Tj [5.89999 ]TJ
(ed. N)Tj [29.60001 ]TJ
(o)Tj [0.10001 ]TJ
( par)Tj [-7.89999 ]TJ
(t of this book may be r)Tj [9.7 ]TJ
(epr)Tj [5.7 ]TJ
(oduced or transmitted in any)Tj
0 -1.1739 TD 0.0168 Tw
(form or b)Tj [5.7 ]TJ (y)Tj [0 ]TJ
( any means, )Tj
```

If individual characters have been kerned by the DTP software, as they all too often are, they will emerge separated by space. Any text between written parentheses may also be deleted.

This is the result after a Replace All. The final mark-up may have to be deleted manually:

```
All rights r eser v ed. N o par t of this book may be r epr oduced or transmitted
in any form or b y any means, Tj
```



## AUTOMATIC HYPHENATION

No system of word-division is perfect and splits may appear to be arbitrary and random. Consider the following examples from the *Oxford Dictionary of Word Division*: such as vegeta-tion but veget-able; pro-vide and pro-vincial, but prov-idence and prov-ince. There are differences between the noun and a verb such as rec-ord and re-cord-ing; prod-uct and pro-ducting; pres-ent and presented You may have a rel-ative who is one of your re-lations; a depen-dant but not depend-ent, and possibly a wan-ton and therefore much want-ed.

There are three varieties of word-division. The American style gives an expectation of what is to follow; the English divides words according to pronunciation; whilst words of Latin or Greek origin separate according to their root. So, you will find at-mos-phere, atmos-phere, or atmo-sphere.

Typeset pages are 'justified' by spacing the words equally along the measure so that the last characters are in vertical alignment on the right. This is done by increasing the normal space unit of one-third of an 'M' slightly, or less desirably, by reducing it. If the inter-word spaces become unacceptably wide, the last word may be split at a suitable point and the remainder carried over.

The hyphenation process examines the over-long word and looks for an existing hyphen to use as a break-point. If there isn't one, an exception list is searched and, if the word is not there, a long list of dividing patterns is read to find a matching group of letters with a splitting point.

However, there are complications. Figures and punctuation, other than dashes and slashes, must be ignored; and all upper case characters converted to lower case before the exception and pattern search can begin. A 'threshold' value indicates the desirable, less desirable, and least desirable splitting places where division may occur.

The patterns are created by consulting authoritative word-division dictionaries such as the Oxford English Dictionary and Webster's. These hold in the region of 60,000 words and, when grouped into similarities, about ten thousand splitting patterns are found. Despite this large number, normal prose only uses in the region of seven hundred combinations; 'in-', 'dis-', and '-ing' and '-ness' being some obvious examples.

Odd numbers in the patterns select divisions in a threshold weighting order where 7 is weak and 1 is strong and even numbers are inserted to prevent any division. A narrow text column using threshold 1 would split ac-cord-ing-ly using the patterns (.ac1) (4d3ing) (1ly.), but a wider book measure, using threshold 3, would divide it as 'accord-ingly'. The even number 4 before 'ing' prevents words like 'mend-ing' or 'send-ing' being split incorrectly as 'men-ding' and 'sen-ding'.

Similarly (.re4al3) and (.re4ad3) avoid a division in those combinations of letters which the prefix pattern (.re3) would otherwise create, whereas (.re5ad4just) enables it again. A true five overrides a false four, which in turn negates a true three. The point or period shows the start or end of a word.

Names of places and people that do not conform to recognizable patterns are placed in an exceptions dictionary and suitable divisions are indicated by a 9 to over-ride the lower threshold numbers 1, 3, 5, and 7 in the patterns.. Words like (.ghost.) or (.through.) inserted without a 9 will never be split.

Commercial word-division dictionaries are jealously guarded by their manufacturers, but one which is freely available was published by Stanford University Department of Computer Science in 1983. Frank M. Liang developed an algorithm of some 8500 patterns for most of the legitimate places to break words in roman languages. These patterns were used by Professor Knuth for his TeX program for scientific typesetting and have been adapted by Olavi Sakari for PostScript Markup. The UK and USA versions are available from <http://home.ricochet.com/osakari>.

The Cappella Archive Tinydivi dividing dictionary used in these pages consists of pre- and suffix combinations instead of the TeX probability algorithms and users may add their own patterns and exceptions. Automatic hyphenation doubles PostScript interpreter time, so a marked-up file should be previously distilled into the portable document before printing.

For the TinyDivi or all you need to know about word-division see:

<http://www.cappella.demon.co.uk/dividing/tinidivi.html>.

<http://www.hyphenologist.co.uk>.



## THE TINYDICT SYNTAX

### Text Mark-up

% The upper case codes need a space before the final text delimiter

#### 1st - 1st

opens a series of columns whose number and width is determined by **NC**. Five columns may be in any order, such as **1st 3rd 2nd 5th 4th** but the first must be opened first even if empty. To restore the body text close the columns with **nocols**. The column style of linespacing, font, and size, is defined in **coltext**.

#### C (text) C

centrally justifies text, linewraps, advances a line and page jumps. Insert a space after the first delimiter as well as before the last.

#### c (text) c

centres text and advances a line. There is no linewrap, nor page jump. North American users may change the English spelling in the Tinydict definition.

#### CS (text) colour CS (text) centre colour CS (text) right colour CS

‘colour show’ places coloured text flush left and linewraps. Insert any movement operators needed and add **L** for a line advance.

#### dc n (character) dc

drop capitals: specify drop in linespacing: character: font. Close the last line with **fj** and restore the left margin: e.g.

**7 LG (W) /Times-Roman dc (. . . last line of drop) fj**

**0 LM MT (normal text width . . .**

#### EN - EN

Ends a footnote and restores the body text and current point.

#### fj (text) fj

force justifies spaced characters, a line, or drop cap lines. There is no page jump and it may be used as a text break for 1upA5 pages re-imposed using the format 2upPP. The odd (recto) page number is always the first of each pair.

**%%Page: 1 1** % section and page no.: e.g. 1 1 . . . 1 2, etc

**1 PG 2upPP** % first page no.: open printers pairs

. . . last line of text on page one ) fj % use p if text is not broken

**%%Page: 1 16** % see 2upPP page format

**16 PG newpage** % footer or header page number

(the text of page 16 starts here . . .

#### FN n FN

places a footnote at the foot of the page and raises **BM** by value of *n*. Insert the footnote text *before* the paragraph with the reference number and end the note with **EN** to restore the body text style and position.

#### inf (text) inf

scales the current font to 70% and places inferior character/s. Any following space should be after the next delimiter.

(H) **s (2) inf (SO) s (4) inf ( is sulphuric acid. ) s** = H<sub>2</sub>SO<sub>4</sub> is sulphuric acid.

#### J (text) J

justifies text flush left and right, advances a line, linewraps, and page jumps. May be used with single words, lines or paragraphs to proof an entire book. See the Grep resource.

#### j (text) j

places text flush left and right, linewraps, and page jumps. There is no line advance and breaks justified text for a change of font, style, or size. Add **L** for a line advance. Use **CS** for colour changes.

*9 rom (This is 9pt roman, ) j 9 it (and this is red italic. ) red CS 8 ss (Include 8pt sans serif, ) j L*

This is 9pt roman, *and this is red italic.* Include 8pt sans serif,

#### LS n (text) LS

centres text and letter spaces the characters with the value of *n*.

#### M (text) M

as **J**, justifies text flush left and right, advances a line, linewraps, and page jumps but also inserts a first line em indent.

#### n (text) n

places text flush left and ragged right and advances to the next line. There is no linewrap but does page jump. Use for the last tab on a line and for across column headers.

#### nj (text) nj

as **J**, justifies text flush left and right, advances a line, linewraps, but no page jumping (hence **nj**): use for footnotes or text that must remain on that page.

#### P (text) P

places text flush left, advances a line, linewraps and page jumps. Use for ragged right text in single words, lines, or paragraphs.



**p (text) p**

places text flush left, advances a line, linewraps, but does not page jump. Use for footnotes or any text that must remain on that page.

**r (text) r**

places text flush right and ragged left and advances a line. There is no linewrap and no page jump.

**S (text) S**

like P, places text flush left and ragged right, linewraps and page jumps. It may be used like j to break text for a change of font, style, or size. Add L for a line advance

**s (text) s**

places text flush left. There is no line advance, no linewrap, nor page jump. Use for all tabs except the last on the line, and for left handed lists with page numbers set flush right with r. A space is not needed before the last delimiter.

**sw n sw (text) w**

adjusts the space widths of the text between the delimiters by the plus or minus value of n. The text must be closed by the **widthshow** abbreviation **w** and an **L** line advance if required. A minus value will tighten spaces.

**5 sw (These spaces are 5 points wide.) w** These spaces are 5 points wide.

**sup (text) sup**

scales the current font to 70% and places superior character/s. Note the use of **s** to avoid a space between the end of the text and the superior figure. Following text may need to start with a space.

(A footnote figure.) **s (1) sup** A footnote figure.<sup>1</sup>

**ul (text) ul**

underlines the text. There is no space before the closing delimiter but one should open the following text.: e.g.

(Never ) **s (underline) ul ( unless adding figures. ) s (25.75 ) ul L**

Never underline unless adding figures. 25.75

indexword 70 **XX ( . . . indexword ) j (indexword ) XX (remaining text . . .**

places a word chosen for indexing in the recto or verso margin with its page number. Close the text with **j** or **S** after each chosen word and repeat the word in delimiters followed by **XX**. The numbering may be removed for hanging margin notes.

**Movement**

**a n a**

advances vertically a specified distance in points down the page.

**A - A**

advances vertically half the linespacing **LG** down the page.

*Replaces H now redefined as a halfnote in the TinyClef music resource.*

**b n b**

reverses vertically a specified distance in points back up the page.

**B - B**

reverses vertically half the linespacing **LG** back up the page.

**em - em**

Moves the current point one 'em' horizontally. Traditionally, it was the width of the upper case **M** in the current font. In the Tinydict it is proportional to the linespacing. Substitute any preferred value but, once chosen, do not alter. *Any change is likely to upset the page breaks and hyphenation in files marked-up previously.*

**en - en**

Moves the current point one 'en' horizontally. Use for the correct alignment of figures which are usually one en wide: e.g.

**en (5) s 5 en en (2.25) s 2.25**  
**(25) s 25 (300.125) s 300.125**

**k n k**

Moves the current point horizontally by the value of n. To move left for kerning use a negative value.

**L - L**

advances by one linespace **LG** down the page.

**MT - MT**

moves one tenth of a point to newline. Use to alter a margin.

**pivot n pivot**

rotates n degrees anti-clockwise at the current point. For clockwise rotation use a negative value. Sandwich it between a **gs - gr** pair to isolate the rotation and use **s** to place the text. Notice the difference between a pivot in advance and one after normal text.

**gs (some text ) 20 pivot s gr (some text ) s** some text  
**(some text ) s gs (some text ) 20 neg pivot s gr** some text some text





## R - R

reverses by one linespace **LG** up the page.

## T n T

tabulates across the page. Use **s** and then **n** for the last one: e.g.

**60 T** (*some*) **s 120 T** (*text*) **s 180 T** (*last*) **s 240 T** (*one.*) *n*  
                  some                  text                  last                  one.

## v n v

Moves the current point vertically by the value of *n*. To move down the page use a negative value and remember to come back again:

e.g. (*who*) **s -2 v** (*o*) **s 2 v** (*ops!*) **s** = *who*<sub>ops!</sub>

### Store marks and Text Manipulation

## HS - HS

stores the current horizontal position: use before the vertical tab **VT**

## HY - HY

selects the threshold strength of hyphenation using odd numbers from 1 (strong) for narrow columns to 7 (weak). The default is 5.

## IN n IN

specifies an inset or outset value. The 0 default is restored after **LB** or **FB**

## hyphensoff - hyphensoff

turns off automatic word division

## hyphenson - hyphenson

turns on automatic word division

## INSET - INSET

insets text or boxes to left and right by a value given to **IN**

## OUTSET - OUTSET

outsets text or boxes to left and right by a value given to **IN**

## SB - SB

starts a box before **CB** or **LB**. Use **B SB** after an **FB** to outline a filled box.

## TI - TI

text inset by the value of half the linespacing **LG**

## T0 - T0

text outset by the value of half the linespacing **LG**

## VS - VS

stores the current vertical position. Starts a vertical rule, column, or frame.

% 72 %

% The Tiny Guide

## Rules, Outlines and Fills

## CB - CB

closes a black outline 0.2 pts wide. Type **SB** at the starting point.

## FB n colour FB

fills a box with colour. Specify drop in lines and colour and place before any text or a line box **LB**. The start box **SB** is not needed.

## FC n colour FC

fills a circle with colour. Specify radius and colour. A tabbing value of half the text width **RM** will centre the circle on the page: e.g.

**rm 2 div T 40 red FC**

## FR n colour FR

frames the entire page below the current point. Specify linewidth and colour.

## frame - frame

frames an imported EPS image. Prefix **place** with **VS** and restore the Tinydict with **text before using frame** and then add the line width and the colour of the box:

*(dir:folder:image.eps)* **VS 50 75 1 place text frame 1 red LB**

## HR - HR

places a horizontal rule 0.2 pts wide or specify width using *n rule*

## LB n colour LB

closes a coloured outline. Specify the linewidth and colour and type **SB** at the starting point.

## LC n n colour LC

places a circular outline. Specify linewidth, radius and colour. A tabbing value of half the text width **RM** will centre the circle on the page.

## LR - LR

places a left-handed vertical rule 0.2 pts wide. Type the **VS** vertical store at the starting point.

## RR - RR

places a right-handed vertical rule 0.2 pts wide. Type a vertical store **VS** at the starting point.

## rule n rule

places a horizontal rule of specified width.

% PostScript Markup

% 73 %



## Font Selection

% The font, size, and style may be changed at any time %  
% but will revert to the current bodytext selection on a newpage.%

### **bol** *n* **bol**

selects the bold typeface in the current textbox.

### **bolit** *n* **bolit**

selects a bold–italic typeface in the current textbox.

### **cr** *n* **cr**

selects the Courier typeface in the current textbox.

### **F** *fontname* **F**

specifies an alternative typeface and size: e.g. **9/Sonata F**  
Level 1 users should uncheck the ‘findfont’ version.

### **it** *n* **it**

selects the roman italic typeface in the current textbox.

### **rom** *n* **rom**

selects a roman typeface and size in the current textbox.

### **sb** *n* **sb**

selects the sans–serif bold typeface in the current textbox.

### **sbi** *n* **sbi**

selects the sans–serif bold–italic typeface in the current textbox.

### **si** *n* **si**

selects the sans–serif italic typeface in the current textbox.

### **ss** *n* **ss**

selects the sans–serif typeface in the current textbox.

## Page and Textbox Defaults

% alterations to margins need MT for immediate effect: e.g. **20 LM MT**

### **BM** *n* **BM**

bottom margin value; default 0. The value is adjusted for footnotes by **FN**

### **FM** *n* **FM**

footer margin value. The distance from the edge of the paper to the bottom margin of the textbox. The default is 72 points (one inch) Use any value but variations may affect the guillotine trimming settings.

### **IM** *n* **IM**

the inner margin of each facing page. The default is 42 points (five eighths of an inch) from the centre fold to the text on either side, so the recto and verso facing pages are 84 points apart. Reduce to 36 points on US letter paper.

### **LM** *n* **LM**

left margin value; default 0. Use *n* **LM MT** to shift the margin across the page, for example, for poetry. Also see **INSET** and **OUTSET**.

### **NC** *n* **NC**

alters the number of columns to a maximum of 5. The default is 2.

### **OM** *n* **OM**

an outer margin value for narrower thumb margins on brochures and news–sheets. The default is 30 points.

### **PG** *n* **PG**

selects the first or an alternative page number. The default is 1.

### **PH** *n* **PH**

the height of the paper being used in points.

### **PW** *n* **PW**

the width of the paper being used in points. Determines the tiling of multipages and the placing of the textbox on the page.

### **RM** *n* **RM**

right margin value. This is the distance across the textbox from left to right. Traditionally, on a book page it was 288 points (24 ems) and known as the ‘measure’.

In the Tinydict it is set at 300 points as a few points are lost in the linewrapping. Substitute any preferred value but **RM** must also be quoted with the paper sizes as well as the margin defaults, as it determines the position of single and verso textboxes. Users of letter paper may prefer to set it at 280 points to provide wider thumb margins.

### **TM** *n* **TM**

top margin value. This is the height of the text box measured upwards from the bottom margin. In the Tinydict it is set at 470 points (6.5 inches) for a book page, but substitute any preferred value.



## Text and Numbering Styles

### **bodytext** - **bodytext**

sets the default font, size, style, and linespacing for the text. Named groups of fonts may be inserted; see section 36.

### **coltext** - **coltext**

sets the default column text font, size, style, and linespacing defined in the page format. Grouped typefaces may be inserted; see section 36.

### **folio** - **folio**

insert **number folio** in ‘numbering’ to centre a folio page number in the footer margin. Type a roman page number for any preliminary pages *after* a newpage : e.g. **newpage (viii) folio**

### **footers** - **footers**

places a running text alternately to the left or right of a central folio page number in the footer margin of recto and verso pages. Insert in ‘numbering’ and type the text in ‘chapter’ and ‘title’ in the page format.

### **headers** - **headers**

places a running text alternately to the left or right of a recto or verso page number in the page header. Insert in ‘numbering’ and type the text in ‘chapter’ and ‘title’ in the page format. Uncheck **HR** in the definition for a header rule.

### **LG** *n* **LG**

sets the width of the linespacing in **bodytext** or any other style setting. The value may be changed at any time but will revert to the current style for each newpage. It also sets the field depth in the Tiny Tables resource and vertical proportions of the Equation Resource.

### **numbering** - **numbering**

insert ‘headers’, ‘footers’, ‘number folio’ between the braces, or leave empty.

### **QT** - **QT**

a quoted text alternative for **bodytext**. The style becomes the default until the body text is restored with **QQ** .

### **QQ** - **QQ**

immediately returns to the **bodytext** from an alternative style.

### **textbox** - **textbox**

resets the bodytext font, size, style, and linespacing for each new page.

## Page and File Management

### **back** - **back** page format

prints the second side of 2upA4 and 4up tiled pairs, or the second side of the first sheet of 16 page nested sections.

### **close** - **close**

prints the last page; removes the Tinydict; restores printer defaults.

### **front** - **front** page format

prints the first side of 2upA4 and 4up tiled pairs, or the first side of the first sheet of 16 page nested sections.

### **gs** - **gs**

preserves existing condition and isolates any following changes in colour, rotation, translation, and scaling.

### **gr** - **gr**

restores the condition previous to **gs**.

### **headright** - **headright**

rotates the text for facing book pages. Places the zero co-ordinates at the leading edge of the paper on A4 printers.

### **headleft** - **headleft**

rotates the text for facing book pages. Places the zero co-ordinates at the leading edge of the paper on long edge input A3 printers.

### **import** - **import**

imports other Tinydict marked-up files. Needs a Distiller filepath. See **useDistiller** and **popDistiller** . The **text** replacement is not needed.

(MacintoshHD:bookfiles:03mylife.ps) import [return]

### **inner** - **inner** page format

prints the first side of the second sheet of 16 page nested sections.



**newpage - newpage**

forces a move to the next page.

**nocols - nocols**

closes the current column and restores the body text. Use before changing the number of columns if in a column: e.g. **nocols 3 NC 1st**

**noDistiller - noDistiller**

for laser printing a file that includes Distiller filepaths, or remove **import** from the **place** procedure.

**outer - outer** page format

prints the second side of second sheet of 16 page nested sections.

**place n n n place**

places EPS graphic. Insert width–height–scale. Distiller needs a file path *before* **place** and the DSC IncludeResource needs one *after* it.

**popDistiller - popDistiller**

for printing files which include both Distiller filepaths and Include Resource comments. See **noDistiller** and **useDistiller** .

**rotatepict n n n rotatepict**

landscapes an EPS image which is too wide for the page. Use instead of **place** .

**quarto - quarto**

divides the paper into four pages.

**SS n SS**

sets the lines per inch screen for halftone images. The default value is set at 106 in the **place** instruction in the Tinydict.

**swap - swap**

halves the paper height to become the book page width and the original width becomes the page height.

**text - text**

restores the Tinydict after a **place** instruction and moves the current point to a new text position below the imported EPS.

**useDistiller - useDistiller**

places files using a Distiller filepath. See **noDistiller** and **popDistiller** .

**\_Z - \_Z**

preserves existing condition and isolates imported EPS files.

**ZZ - ZZ**

restores the condition previous to **\_Z**.

% 78 %

% The Tiny Guide

**Page Formats**

In the page formats the text auto–flows to a newpage, unless an earlier page break is specified by **newpage** or the non–page jumping codes **p**, **nj**, or **s L** are used.

**1upA3 - 1upA3**

prints single pages on A3 or tabloid paper for posters or art–work.

**1upA4 - 1upA4**

prints single pages on A4 or letter paper for correspondence.

**1upA5 - 1upA5**

produces single pages on A5 or half–letter as PDF e–books or for on–screen proofing. After distilling, export from Acrobat Exchange as DSC compliant pages for imposition into commercial multipage sections and signatures.

**2upA3 - 2upA3**

prints two identical sets of four page folios on A3 or tabloid paper. Prefix **front** and **back** to print the first and second sides.

**2upA4 - 2upA4**

prints facing pages on A4 or letter paper for folding into four page folios. Prefix **front** and **back** to print the first and second sides.

**2upLR - 2upLR**

produces facing pages for left to right reading for PDF e–books or on–screen proofing.

**2upPP - 2upPP**

enables facing pages for multi–page sections. Pages are selected in correct order by a control file using Distiller filepaths or a document manager DSC Include Resource facility. Alternatively, individual page files may be file–merged, or pasted by hand in sequence using **fj** at the page breaking points. Each page needs its correct **PG** number before the page format and a newpage instruction. See **fj** syntax.

**4upA4 - 4upA4**

prints half–size facing pages on A4 or letter paper for proofing. Prefix **front** and **back** for work and turn folio pocket books.

**8booklet - 8booklet**

prints eight pages in correct order on both sides of A3 or tabloid paper for folding into stapled programmes or instruction booklets. Prefix **front** and **back** to print first and second sides.

% PostScript Markup

% 79 %



**8wide - 8wide**

prints four landscaped pages in correct order on both sides of A3 or tabloid paper for folding into anniversary cards. Prefix **front** and **back** to print first and second sides.

**16nested - 16nested**

prints sixteen pages in correct nested order on both sides of two sheets of A3 or tabloid paper for folding into sections for stapled booklets or quality sewn bookwork. Prefix **front** and **back** to print the first and second sides of the first sheet and **inner** and **outer** similarly for the second.

**SOME POSTSCRIPT UTILITIES****Software suitable for Direct PostScript Typesetting**

A line unwrapping text editor is essential for scrolling through hundreds of pages of book copy quickly. The unwrapping removes the PC line feed double space and the Tinydict mark-up codes may be inserted automatically by a prefix and suffix macro.

**Line unwrapping text editor for PCs**

NoteTab Pro 4.5: <http://www.notetab.ch>:

Opens documents up to 16 Mb. Tabbed layout and customizable toolbar. Spell checker, thesaurus, text to HTML, sorts lines, etc.

**Line unwrapping text editors for Macintosh**

The following are usually available on Mac magazine CDs. BBEdit Lite. Very useful PostScript and pre- and suffix Extensions Tex-Edit. Good search and replace.

Alpha. Needs large disc space; libraries confusing to a novice.

**PostScript printing or viewing for PCs**

PrintFile: <http://www.lib-sys.com>: freeware utility for Windows

PSalter 1.5: <http://www.quite.com>: good PostScript viewer and editor

PSprinter: <http://www.adobe.com>: the Adobe PostScript downloader

**Postscript printing or viewing for Macintosh**

Let erRIP: <http://www.lupinsw.com>:

A document manager with Include Resource facilities. Do not confuse with LetterRIP .

PS2EPS+: <http://www.lib-sys.com>:

Freeware: PS conversion to EPS, TIFF and PICT. Prints good images but poor text. Sits on top of a useable MacGS 2.0

EPStoPICT: <http://artage.com>:

Converts PS or EPS to PICT and rasterises up to 350 dpi. Files view in SimpleText, print on non-PostScript printers, or import into PhotoShop. Level 1 but uses ATM so good text.

PSprinter: <http://www.adobe.com>: The Adobe PS downloader

Drop.PS: Mac ftp sites

ShowPages: Mac ftp sites:

Freeware printer driver: selects paper sizes and works with emulating software.



## Portable document format utilities

Reader: <http://www.adobe.com/prodindex/acrobat/readstep.html>: Adobe freeware for reading and printing existing PDF files.

PDF995: <http://www.pdf995.com>: Windows freeware for PDF.

PDFCreator: <http://www.jawspdf.com>: Mac shareware for PDF files

FormView: <http://www.pcltools.com/formview.htm>: freeware HP PCL to Acrobat PDF.

FreeDist: <http://home.hccnet.nl/s.vd.palen>: PC freeware converter.

txt2pdf 6.4 version: <http://www.sanface.com/txt2pdf.html>:

Perl5 script that converts text files to PDF format.

PDF PageMaster: <http://www.pdftron.com>:

PC software to split, assemble, merge, and manage PDF files

Ari s Print Helper: <http://www.dionis.com/pdf.html>:

Mac Acrobat plug-in to print PDF pages in any order

XPDF: <http://www.foolabs.com>:

PDF converter for Unix, Acorn, BeOS and Amiga.

## Other PostScript Utilities

For those banging their heads against the digital wall see:

<http://www.geocities.com/siliconvalley/5682/postscript.html>

<http://www.wiskit.com/postscript/quicktips/>

<http://www.this.net/~frank/>

<http://www.acumenjournal.com/>

Practical PostScript: A Guide to Digital Typesetting

<http://www.cappella.demon.do.uk/bookpdfs/pracpost.pdf>

## Commercial applications

Streamline: Adobe: Converts TIFF files to PS vectors.

Acrobat: Adobe: Essential for serious e-book work.

Tailor 2.0: Enfoc Software: <http://www.enfocus.com>: PDF editor.

## PRINTING A BOOK

### The Copy

*Observe some good typographical practice:*

1. No text formatting. No centering with the space bar or tabbing.
2. Only return paragraphs or single lines: no hyphenation.
3. Unless facts, write figures under 100: fifth century: 1200 dollars
4. Only *one* space after a point (a.k.a. full stop or period).
5. In PS encoding, use the tilde key for left single curly quotes.
6. Use the straight apostrophe for right hand single curly quotes.
7. Use Mac opt-2 and opt-b for left and right hand double quotes.
8. No plural apostrophes unless single letters: 1900s: p's and q's.
9. Spell check, putting unusual words in the dictionary first.
10. Remember difference between its/it's: their/there: who's/whose.
11. Check recogn-ise or -ize, equalize, etc. in the OED or Webster's.
12. Save as .TXT in the dialog: do not save as .RTF, MSW, etc.
13. Make a back-up reference copy.

### Typesetting

1. Open the copy with a text editor: NoteTab, BBEedit, etc.
2. Use 'M' to indent the first line. OR Increase paragraph spacing by adding, say, '3 a' in the P or J codes. (Do not do both.)
3. Use Find-Replace to prefix any ( and ) with a \ (backslash).
4. Unwrap the text so it runs off the right hand of the screen.
5. Prefix every paragraph with a left hand (
6. Suffix every paragraph with a space and ) and P or J or M
7. Remove any ( ) P or J or M from empty lines.
8. Change P or J to c or r for centered or right handed lines of text.
9. Insert EPS 'place' values but do not import into position.
10. Type 'newpage' as required, 'close' at end: proof with 4upA4.
11. For sections (not folios) cut and paste into 2upPP pairs.  
(Use 'fj' to force justify the last line of split paragraphs.)
12. Distil, filemerge, or paste, the EPS files into position.
13. Print as '2upPP' for booklets or '2upA4' for sequential folios.

### Binding

See page 89





## THE SELF-PRINTING FILE

Paste a copy of the Tinydict at the top of a new file.  
Insert your chosen page format, 1up, 2up, 4up, 8up or 16up.  
Substitute any running header and footer text.  
Type 'headers' or 'footers' in 'numbering' or leave empty.

### Laser printing:

*If possible use the manual or multi-purpose tray.*

For folios use 'front 2upA4' or 'front 4upA4' for proofing.  
For the reverse use 'back 2upA4' or 'back 4upA4'  
For printer's pairs use '2upPP' for front page file (1:16 etc)  
For the reverse use '2upPP' for back page file (15:2 etc).  
Back and front toggle is not needed for 1up or 2up formats.  
Download non-resident laser fonts before printing.  
Laser print with a PS downloader or software emulator.  
Adjust the 'translate' value to align front and rear printing.  
Use headright landscaping for short edge feed printers.

### Distilling PDF pages:

For e-books use '1upA5': or '2upLR'.  
The Tinydict and files should be in the same folder.  
Check that chosen fonts are available to Distiller.  
Do not include 'manual', 'normal' or 'copies' in Distiller files.  
Download the Tinydict with the 'startjob' unchecked.  
Ignore the message "No PDF file produced".  
Open the marked-up files from the same folder and distil.  
Alternatively, use the 'RunFile' facility as explained below.

### Importing marked-up pages or EPS images:

Insert by typing 'place' at the relevant point; see Section 15.  
There are four methods:

- a) Type the RunFile filepaths **before** 'place' if using Distiller.
- b) Type the Include Resource filepath for a document manager **after** 'place'.
- c) Filemerge them into a **copy** of the typeset page file **after** 'place'.
- d) Cut and paste them by hand at the relevant point.

Previously typeset Tinydict files are positioned in Distiller by using the 'import' instead of 'place' and do not need the 'text' replacement afterwards.

## CHOOSING PAPER

Paper is produced from a continuous stream of pulp, which is gradually dried, smoothed, and rolled up into a large cylinder. This horizontal process produces a 'grain' in the paper, with the fibres lying with the direction of conveyor travel. As a result, the paper is stronger under lengthways tension and suitable for feeding off the roll onto large presses.

When the paper is cut into successively smaller printing sizes, the direction of the grain changes alternately from long to short, which is across the width. So A0 is long, A1 short, A2 long, and A3 short. By the time A4 and North American letter sizes are reached, the paper is once more long-grained, which allows letters and report pages to be turned over easily.

You may discover the grain direction for yourself by taking a piece of A4 or letter paper and running two adjacent edges quite hard between your thumbnail and first finger. You will find that shorter side wrinkles, showing that the paper is 'long-grained', and as a result folds and tears more easily lengthways down the corrugations than it does crosswise.

Unfortunately, if you print folio book pages using off-the-shelf long-grained A4 or letter size paper, the folded sheets will never open completely flat, nor close completely. There will also be a crackling noise as the pages are turned over and the spine will distort using a water-based glue. In fact, it will behave just like many commercial paperbacks, when the grain of the paper runs across the folded book pages instead of up and down.

To make a better book, take 25 sheets of A3 or US tabloid paper and cut them 8.5 inches (216 mm) wide down the length of the A3, so as to be 'short-grained' across the paper. Reduce the length to 11.7 inches (297 mm).

The letter width makes better proportioned pages in combination with the longer A4 length, which in turn prevents gripper smear. To avoid paper creep, only cut four pieces at a time; if you doubt your dexterity, any local printer will guillotine them for you. An accurate length is more important than width. When printing this size, North American users should remember to choose A4 paper in the page setup dialog.



## PRINTING FOLIO PAGES

How you print folio pages depends on your printer. You may:

1. Print one sheet at a time, turning over to print the reverse.
2. Print all the odd paired pages; then the even pairs on the reverse.
3. Duplex print the all the paired pages.

Laser printers invariably produce pages printed side down; on the first pass turn each one over as it emerges, so that the blank side of page 1 is on the top of the stack.

If duplexing, select short edge, but do try two sample pages first as there is sometimes a difference in back to front registration. If there is a discrepancy, use odd/even printing.

Only print a two pages at a time pages on an inkjet printer, unless your computer has large memory and plenty of disc space for the file to spool properly.

## BINDING INSTRUCTIONS

1. Omitting the first, fold the sheets with spine mark on the outside.
2. Firm the creases with a pen top or back of a comb.
3. Cut two pieces of card and waxed paper or acetate to page size.
4. Stack the sheets in order, again omitting the first titled sheet.
5. Place the waxed paper (wax inwards) or acetate either side.
6. Add a card to either side, secure lengthways with an elastic band.
7. Jog level at base and folded spine, glue the folds with PVA.
8. Leave to dry under weight. Remove card and waxed paper.
9. Wrap the title page round book and align the spine text.\*
10. Remove; crease folds; glue spine, attach, and trim when dry.

\*The title page may be printed on some coloured paper or thin card.

## REVISIONS v 7.0 – NOVEMBER 2004

Letter spacing code 'LS' added.  
Automatic Distiller and printer toggle.  
Automatic Hyphenation added using TeX or TiniDivi Dividing Dictionary.

### REVISIONS v 6.0 – JANUARY 2003

Text font, size, style, and line spacing, combined in bodytext  
Text in and text out TI and TO added  
Page-jumping columns provided  
Outline and filled boxes and circles simplified further  
First line em indent full justification code 'M'  
One tenth of a point movement abbreviation 'MT'

### REVISIONS v 5.0 – APRIL 2001

Duplicated 'show' removed from S and p codes  
Columns resource revised for any paper size  
Equation setting resource  
Superior and inferior codes changed to 'sup' and 'inf'

### REVISIONS v 4.0 – DECEMBER 2000

Full justification procedures made available.  
Outline and filled box setting simplified  
Automatic footnote and section numbering resources.  
TinyImp pdf2book resource for 1up PDF pages to 2up book folios.

### REVISIONS v 3.0 – JANUARY 1999

TinyLinx resource for PDF marks.  
Improvements to INSET and OUTSET values  
Grep search and code insertion for automatic PostScript mark-up.  
Grep single decimal place reduction for vector EPS files.

### REVISIONS v 2.3 – OCTOBER 1998

TinyImp resources for 1, 2, 4, 8, 16 page impositions.  
TinyTables resource with field text justification and basic calculation  
No 'text' instruction needed after a Distiller EPS 'import'.  
Setting ersatz superior and inferior characters.

### REVISIONS v 2.2 – JUNE 1998

Bug removed affecting non-justified text in a 2nd column.  
'nocols' resetter added for book page columns.  
Colour section moved into separate resource.  
Multiple column resource for brochures and newsletters  
Mac fonts re-encoded for ISO Latin-1.

### REVISIONS v 2.1 – MARCH 1998

Coloured text coding and rectangular and circular fills and outlines  
Book page columns and left and right vertical rules.  
Inter-changeable headers and footers.





## Cappella Archive

*Book on Demand Editions*

Foley Terrace : Great Malvern : England

01 684 565 022 : WR14 4RQ

[www.cappella-archive.com](http://www.cappella-archive.com)

### What Others Think

*This handsomely executed limited edition book will make a fine present.*  
New Law Journal

*The production is absolutely superb.*                      The Cloak & Dagger Club

*The book, produced to excellent standards, is a collector's item.*  
The Ripperologist

*The book is attractively produced.*                      Worcestershire Wildlife

*A real joy to behold.*    The Organist's Review

*Exquisitely produced.*    The Church Times

*Beautifully presented,*    Old Chorister Review

*It is actually printed on decent paper, in a clear type, well set, with seemly margins.*  
Barry McKay Rare Books, Appleby, Westmorland

**Cappella Archive Editions are all typeset by  
the Tinydict PostScript Mark-up in a non-kerning  
digital version of a Fry s Baskerville of 1769.**

**For a Synopsis of the 236 character combined roman-italic typeface see:  
<http://www.cappella.demon.co.uk/tinypdfs/baskcomb.pdf/>**

**For the current publications catalogue see:  
<http://www.cappella.demon.co.uk/cappubs.html/>**